Graph-Based Approach on Social Data Mining

BY

GUAN WANG
B.S., University of Science and Technology of China, China, 2007

THESIS

Submitted in partial fulfillment of the requirements
for the degree of Doctor of Philosophy in Computer Science
in the Graduate College of the
University of Illinois at Chicago, 2014

Chicago, Illinois

Defense Committee:

Philip S. Yu, Chair and Advisor
Bing Liu
Brian Ziebart
Daniel Tunkelang, LinkedIn
Chen Chen, Google

UMI Number: 3668648

All rights reserved

INFORMATION TO ALL USERS
The quality of this reproduction is dependent upon the quality of the copy submitted.

In the unlikely event that the author did not send a complete manuscript
and there are missing pages, these will be noted. Also, if material had to be removed,
a note will indicate the deletion.

UMI 3668648

Published by ProQuest LLC (2014). Copyright in the Dissertation held by the Author.

Microform Edition © ProQuest LLC.
All rights reserved. This work is protected against
unauthorized copying under Title 17, United States Code

ProQuest LLC.
789 East Eisenhower Parkway
P.O. Box 1346
Ann Arbor, MI 48106 - 1346

This thesis is dedicated to my family.

ACKNOWLEDGMENTS

First and foremost I would like to thank my Ph.D. advisor Professor Philip S. Yu. I deeply appreciate his support along the way. His mentoring and patience is my biggest wisdom gain during my graduate studies and research at the University of Illinois at Chicago. I am truly indebted to him for providing me with complete flexibility to play with a wide range of research topics based on my pure interests throughout my Ph.D. research. Without his guidance and advising, this thesis would never have been accomplished. My gratitude also go to Dr. Bing Liu. It is my honor to work with him and he always gave me generous help. His inspiration and the way he deals with research has a very positive impact on my research.

I would like to thank my committee members Dr. Brian Ziebart, Dr. Daniel Tunkelang, and Dr. Chen Chen, for their valuable advice, suggestions and time. I also want to thank Linkedin for providing me opportunities to apply my research into practice.

I wish to also express my warmest thanks to all my colleagues in the Big Data and Social Computing (BDSC) Lab at the University of Illinois at Chicago. I am very grateful to work with them during my graduate studies and there are countless fun and enjoyable moments while we discuss and learn.

Lastly, I would like to thank my family for their support, trust and encouragement. I want to give my special gratitude to my parents and Xia for their unconditional understanding and support. Without them, not only there would not be this thesis, but also there would not be this version of me with any achievements in my life.

GW

TABLE OF CONTENTS

TABLE OF CONTENTS (Continued)

CHAPTER **PAGE**

TABLE OF CONTENTS (Continued)

CHAPTER **PAGE**

LIST OF TABLES

LIST OF TABLES (Continued)

TABLE **PAGE**

LIST OF FIGURES

LIST OF FIGURES (Continued)

SUMMARY

Powered by big data infrastructures, social network platforms are gathering data on many aspects of our daily lives. The online social world is reflecting our physical world in an increasingly detailed way by collecting people's individual biographies and their various of relationships with other people. Although massive amount of social data has been gathered, an urgent challenge remain unsolved, which is to discover meaningful knowledge that can empower the social platforms to really understand their users from different perspectives.

Motivated by this trend, my research addresses the reasoning and mathematical modeling behind interesting phenomena on social networks. Proposing graph based data mining framework regarding to heterogeneous data sources is the major goal of my research. The algorithms, by design, utilize graph structure with heterogeneous link and node features to creatively represent social networks' basic structures and phenomena on top of them.

The graph based heterogeneous mining methodology is proved to be effective on a series of knowledge discovery topics, including network structure and macro social pattern mining such as magnet community detection (87), social influence propagation and social similarity mining (85), and spam detection (86). The future work is to consider dynamic relation on social data mining and how graph based approaches adapt from the new situations.

CHAPTER 1

INTRODUCTION

1.1 Dissertation Framework

Social data are accumulated rapidly with the prosperous growth of social network services. The user generated content, such as relations, shares, blogs, comments, are reflecting people's real life experiences on many kinds of social network platforms, including Facebook, LinkedIn, Twitter, Tumblr, Pinterest, Foursquare, and so on. Such social data grant researchers the new opportunities to understand a whole spectrum of new phenomena related to users. For example, data scientists could discover hidden and promising communities, identify influential people in the vast social media population, infer dynamic or static relations, etc. The capability to make reasons out of social data becomes critical for many aspects in data mining tasks, including decision making, business intelligence, computational advertising, recommendation system, as well as data mining methodologies, e.g., clustering, classification, ranking, with social flavor.

Due to the rich structural and content information of social data, it is necessary to represent them with more suitable forms than transactional. Graph is a data structure consisting of vertices and edges. It is the best-suited mathematical representation for such complex data scenarios that comprise multiple objects and their inter-relationships. Heterogeneous graphs contain even more than one type of nodes and links, which is ideal for social data representation.

Graph formulation can not only be suitable representation for social data, but also novel solutions for social data mining problems. Motivated by this promising direction, we have conducted several new research topics and proposed graph-based techniques in this thesis. We first invented the review graph as an instance of the graph based approach to tackle the online shopping review spam detection problem (86). For example, in the online store review domain is modeled as a heterogeneous graph, where reviews, users and stores are different type of nodes and their belonging relations are edges. From this graph, one can derive trustiness for users, reliability of stores, and honesty of reviews. It is the first time that such spammer detection is done without any review text information. Extending the graph based approach, we studied in (85) the social influence and social similarity reinforce relation, which is modeled together for the first time, via a heterogeneous graph. The two research areas are important and flourishing on their own. However, they reinforce each other and make each other better, if computed properly. The graph model through heterogeneous network once again reveal such reinforcement and empowers the modeling intuitive and reasonable. Furthermore, we proposed magnet community mining project (87) and identified communities of such property within a given network structure. The graph representation and modeling reveals the property of magnet for communities. This property depicts how the community is attractive to members from other communities, how the attraction sustain, and strong enough to attract people's attention from other high quality communities. Finally, we apply the graph formulation and reinforce computation to the social hierarchy mining problem. To be specific, we reveal the organization chart structure within professional social network.

To conclude this section, Figure 1 displays the overall framework of this thesis.

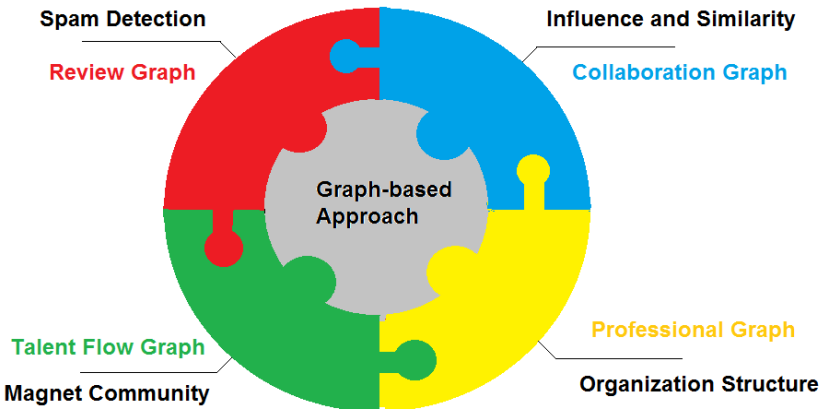


Figure 1. Thesis Framework

1.2 Graph based Approach to Online Store Spammer Detection

Online shopping reviews provide the valuable information for customers to compare the quality of products, store services, and many other aspects of potential purchases. However, spammers are joining this community trying to mislead consumers by writing fake or unfair reviews to confuse the consumers. Previous attempts have used reviewer's behaviors, like text similarity and rating patterns, to detect spammers. These studies are able to identify certain types of spammers, e.g., those who post many similar reviews about one target. However, in reality, there are other kinds of spammers who can manipulate their behaviors to act just like normal reviewers, and thus cannot be detected by the available techniques.

In this thesis, we design a novel concept of review graph to capture the relationships among all reviewers, reviews and stores as a heterogeneous graph. We explore how interactions between nodes in this graph could reveal the cause of spams and propose an iterative computation model to identify

suspicious reviewers. In a review graph, we have three kinds of nodes, namely reviewer, review, and store. We capture their relationships by introducing three fundamental concepts, the trustiness of reviewers, the honesty of reviews and the reliability of stores, and identifying their inter-relationshihps. One reviewer becomes more trustworthy by having written more honesty reviews. One store becomes more reliable by having more positive reviews from reviewers who are trustworthy. One review becomes more honest if many other honest reviews share the common idea with it. We develop an effective computation method based on the proposed graph model and we use no review text information. Our model is complimentary to existing approaches and able to find more difficult and subtle spamming activities, which are agreed upon by human judges after they evaluate our results.

1.3 Influence and Similarity on Social Networks

In social network research, the studies on social influence maximization and entity relevance are two important and orthogonal tasks. On homogeneous network, social influence maximization research tries to identify an initial set that maximizes information spreading, while relevance studies focus on designing meaningful ways to quantify entities' similarities. When heterogeneous networks are becoming ubiquitous and entities of different types are related to each other, we observe the possibility of merging the two directions together to improve the performance for both of them. On the one hand, the influences among one type of nodes could be defined more accurately with the consideration of relevance of the other type of nodes because of their inter-connections. On the other hand, the computation for relevance relations of the other type of nodes could also be more informative with the input of the influences of the former type of nodes.

Therefore, we introduce a framework that computes social influence for one type of nodes in a heterogeneous network, and simultaneously measures relevance of the other type of nodes. Firstly, we decouple the target heterogeneous network (or we call it *Influence Relevance (IR)* network) into three different parts: influence graph (*IG*), relevance graph (*RG*) and information tunnels (*IT*) between *IG* and *RG*. Then through *IT*, we exchange the influence scores and the relevance scores to calculate more precise relevance and influence scores in order to improve both of their qualities. Our mutually reinforced algorithm stops when these scores converge. The experiment results on real world data show that our framework enables influence maximization framework to identify more influential seeds in *IG* and relevance measures to produces more meaningful relevance scores in *RG* simultaneously.

Social network is a popular topic in research community. People have proposed many novel directions of analysis on social network to explore its properties of many different aspects, e.g., structure and information spreading flow. Two prominent techniques of such analysis are *social influence maximization* and *entity relevance analysis*. Given a user population and their relations in an information cascade senario, social influence maximization is centered on constructing a meaningful influence network, identifying a set of most influential nodes in the network, and maximizing the spread of influence through such a social network. It enriches our knowledge about dynamic information handling behaviors of nodes in social networks and helps us design applications such as virtual marketing, recommendations, and information retrieval (44)(80). Relevance analysis, as another flourishing research direction, is proposing methods for measuring nodes similarities, based on the network structure and node features. Knowing relevance relations among nodes gives us tools to perform many fundamental tasks such as clustering or classification, as well as useful applications such as information retrieval,

ranking, or recommendation(75)(94). Although influence and relevance analysis could provide tools for similar applications, they are often considered as orthogonal techniques that are studied separately, besides they are often applied to homogeneous networks. When the heterogeneous network is becoming ubiquitous, we notice that it is not only necessary but also beneficial to combine the two techniques into one framework because we can use information from one side to calibrate the other side.

1.4 Magnet Community Identification

Social communities connect people of similar interests together and play essential roles in social network applications. Examples of such communities include people who like the same objects on Facebook, follow common subjects on Twitter, or join similar groups on LinkedIn. Among communities, we notice that some of them are *magnetic* to people. A *magnet community* is such a community that attracts significantly more people's interests and attentions than other communities of similar topics. With the explosive number of self-formed communities in social networks, one important demand is to identify magnet communities for users. This can not only track attractive communities, but also help improve user experiences and increase their engagements, e.g., the login frequencies and user-generated-content qualities. In this chapter, we initiate the study of magnet community identification problem. First we observe several properties of magnet communities, such as attention flow, attention qualify, and attention persistence. Second, we formalize these properties with the combination of community feature extraction into a graph ranking formulation based on constraint quadratic programming. In details, we treat communities of a network as super nodes, and their interactions as links among those super nodes. Therefore, a network of communities is defined. We extract community's magnet features from heterogeneous sources, i.e., a community's standalone features and its dependency features with

other communities. A graph ranking model is formulated given these features. Furthermore, we define constraints reflecting communities' magnet properties to regularize the model. We demonstrate the effectiveness of our framework on real world social network data.

1.5 Inferring Dynamic Social Hierarchy Relations from Heterogeneous Information Networks

Dynamic nature is an important aspect of many kinds of real world social relations. The inference for such relation, especially with hierarchical information, is under explored. Existing social relation mining mainly targets static relations, or treat dynamic relations as static ones, as we analyzed in related work section. Comparing with static relation mining, there are three unique challenges with dynamic cases. First of all, dynamic nature makes it lack clean-cut rules to follow. Although manager-employee relation mining seems to be a similar task to advisor-advisee mining, all the clear rules in the static case cannot apply to the dynamic case, as it indicates in Table 1. Secondly, the data used for the relation inference may be inconsistent to the current situation. In static case, there is no such a challenge. For example, the papers published by a student and the advisor in the past is a strong indicator of their relation. However, in dynamic case, historical data may mislead the algorithm, since a manager-employee relation in the past may not exist in the present. Thirdly, in a dynamic case, the cleanness of the data is another issue. Take Linkedin social network as an example. People may not update their profile, e.g., job title, time to join a company, in time to reflect the dynamic changes of their working relations. In the static case, such as family relation, there is no out-dated data issue.

OBSERVATION (Dynamic Relation) In social networks, a relation is dynamic *when there is no guarantee that it cannot be changed. There are usually less clear rules to follow than the static relation mining, and there exists inconsistency in historical data to the current snapshot of the dynamic relation.*

The dynamic relation may have hierarchical structure, e.g., manager-employee relation can form orgchart. Due to the challenges, pairwise dynamic relation inference alone is already non-trivial. The more valuable hierarchical structure of such relation is even harder to imply. An analogy of this problem is the Phylogenetic tree construction, where the goal is to construct the evolutionary tree for species using the local inferences of ancestors (23). It is proved to be a NP-hard problem and brute-force method is computational infeasible. In the following subsections, we will derive and integrate features that are suitable for dynamic relation inference and propose an efficient algorithm for the global hierarchy construction.

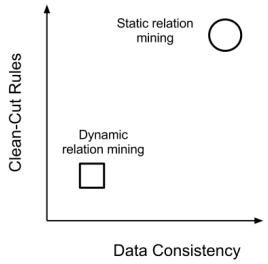


Figure 2. Property Differences of Dynamic and Static Relations

In addition to the dynamic nature, dynamic relations differ from static ones in two major ways. First of all, there lacks clean-cut rules for the relation inference. For example, unlike the publication records for advisor-advisee relation discovery, there is no strong signals for manager-employee relation identification. Table XVII lists the pairwise comparison of rules for an example of dynamic and static

relation mining. In an academic network such as DBLP, there are certain rules that guide us design mining algorithms for the static relation. However, none of these rules work as they are in a professional network, such as Linkedin.

	Advisor-advisee (static)	Manager-employee (dynamic)
connection	academic collaboration	voluntarily and adhoc
time	advisor has longer publication history than advisee	time to join a company does not imply job seniority
title	Professor/Student clear indicator of seniority	"code monkey" or "data guru" adhoc

TABLE I

STATIC AND DYNAMIC RELATION RULE COMPARISON

Consistency of the data is another new challenge for dynamic relation mining. If we only use the most recent data, there may not be enough information. However, if we use historical data, there will be inconsistency information with the current situation, such as company re-organizes itself. Static relation mining does not suffer from this difficulty. For example, historical publication records are indeed favorable information to infer advisor-advisee relation.

The two major challenges make pairwise dynamic relation hard to infer. Construction of hierarchical structure for the dynamic relations, such as orgchart of a company, is a even more challenging problem. Despite the challenges, such relation discovery is of great interest for industries, since many kinds of dynamic relations are greatly valuable. For example, the orgchart mining on Linkedin empowers better

targeting for business to business advertisement, close friend or lover relations mining on Facebook will make the platform optimize its service to the users.

We are motivated by these challenges to design a framework for generic dynamic hierarchy relation mining, as well as implementing a system for one specific type of such relation, i.e., manager-employee relation on LinkedIn social network. First we try to integrate weak signals, e.g., vague connection indications, obsolete profiles, adhoc job titles, from heterogeneous sources to exhibit useful features. We proposed the concept of *objective entanglement group* and *subjective coherence strength* to capture the dynamic relationship from objective and subjective perspectives. Second, these features are used to compute local probability of a dynamic relation. We design a local probabilistic function based on the physics principle of gravity. Finally, global dynamic relation probability is computed through a variation of Markov Decision Process (35).

In summary, our contribution lies in three folds.

- We are the first to tackle the dynamic hierarchy relation discovery problem in social networks. The novelty of the problem lies in two folds. First, dynamic relations on social networks are valuable and neglected by previous research. Second, unsupervised approach for hierarchy construction has never been done in the setting of dynamic relation on social networks. Although a few studies exist (62)(93), they are not applicable to our input and output, as well as rich social network features.

- We discover the connection between dynamic hierarchical relation mining problem and physics principle of forces. We integrate weak features from both subjective user generated social network features and objective opinions to mine dynamic relations to generate an effective local (or

pairwise) dynamic relation predictor. We further formulate the global hierarchy optimization as a reduced Markov Decision Process problem, where the transit between states in a random walk represents the establishment of dynamic relation candidates. We also adopt a customized shortest path algorithm to solve the rMDP problem.

- We do the experiments on real world data sets from LinkedIn social network. We build a system for the mining task from raw data extraction to final result visualization. According to the experiment results on two companies, our framework can produce accurate dynamic relation prediction. Therefore it is promising for real world applications.

CHAPTER 2

REVIEW GRAPH BASED ONLINE STORE REVIEW SPAMMER DETECTION

2.1 Introduction

Online shopping is a major way of shopping for a large number of people nowadays. With the help of latest customers' reviews about products and stores, one can make wise choices for his/her purchases. Due to its important role, the online review system has became a target of spammers. Spammers are malicious reviewers who are usually hired or enticed by companies (2) to write dishonest reviews to promote their own products or services, and/or to distract customers from their competitors. Driven by profits, there are more and more spam reviews in major review websites, such as PriceGrabber.com, Shopzilla.com, or Resellerratings.com(1)(2). And spammers are starting to confuse the consumers.

2.1.1 Challenges

Due to the large volume of online reviews, automatically identifying spammers is an urgent task for e-commerce. However, unlike other better studied spam detection, e.g., email spam (13) or search engine spam (30), review spam is much harder to detect. The reason is obvious: Spammers try their best to disguise themselves as a real reviewers, and it gives human a hard time to distinguish their reviews from real reviewers, not to mention computers; while in other forms of spam detection fields, one can easily tell spams from normal behaviors.

Previous spam fighters use behaviors of reviewers to catch product review spammers, e.g., review text similarities, rating similarities and deviations, the number of spammed products, etc. According to

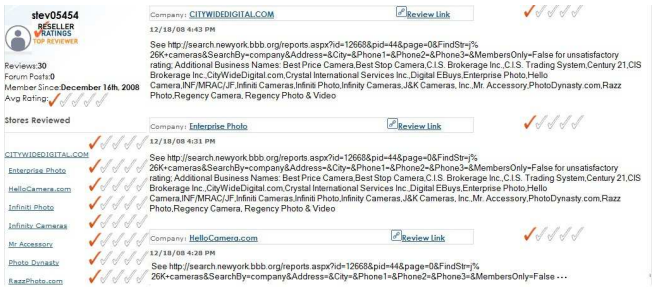
the existing studies, these behaviors contain clues of certain types of spamming activities. For instance, if a reviewer uses a significant amount of similar text in multiple reviews about the same product, or a reviewer rates different products constantly high or low from the same store within a short period of time, (s)he may be a spammer (40)(56), since these activities may imply one type of spamming that spammers try to make the most impact out of the least effort.

Unfortunately, even if we can borrow experiences from previous studies, these clues may not be discriminative or sufficient enough to tell store review spammers from benign reviewers. For example, although it looks suspicious for a person posting repeating reviews to one product, it can be quite ordinary for the person posting many reviews to one store from many purchasing experiences. For example, as one person has the same writing style on review writing, it may be normal to have similar reviews from the same reviewer across many stores or products. Moreover, many ordinary users only write reviews sporadically. It also makes sense for them to write multiple reviews within a short period of time for different shopping experiences. Therefore, reviewer behaviors proposed in the existing approaches may not be sufficient to catch spammers of store reviews. Besides, there are other types of spammers who carefully design their fake reviews in a more deceptive way that would slip through existing spammer detection techniques.

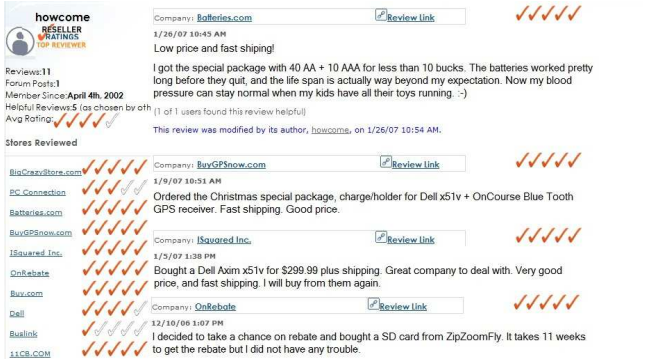
Figure 3 shows the profiles of two reviewers from Resellerratings.com that demonstrate such inaccuracy and insufficiency. The first case is that a reviewer kept posting similar reviews that seem unrelated to the reviewed stores, in a short period of time. These duplications and short-time rush phenomena could be a sign of spamming and they have been used as important clues to catch spammers in (56)(41)(40). However, after manually checking the content of the links plus other evidences gathered

from the Internet, we found that these duplications are posted by a genuine warm-hearted reviewer who tried to warn other consumers about a cluster of lousy stores that have many complaints.

The second case is even more subtle. The person's behaviors may seem normal at the first glance. However, we can be almost sure that (s)he is a spammer if we go beyond behaviors of this single reviewer and consider the stores this reviewer commented on, and other people's reviews of the same stores, and their reviews about other stores. (more discussions in Case Study, Section 4)



(a) seems suspicious, but benign reviewer



(b) seems normal, but really suspicious reviewer

Figure 3. Reviewer Examples

The above examples reveal the incompleteness of existing spammer detection methods and the need to look for a more sophisticated and complementary framework. However, the following challenges are major obstacles towards such a framework.

1. There is *no ground truth* of whether a review is written by a spammer or not. By reading a review text alone, we usually don't have enough clues to tell if it is a spam or not.
2. Spammers' behaviors may be hard to capture. For example, in order to successfully mislead customers, spammers can make their writing styles and review habits look very similar to those of ordinary reviewers, as shown in the second case in Figure 3.
3. A spammer can also write good and honest reviews, because they could be real customers of some online stores themselves some times. Even more complicated, a current good reviewer could be a spammer before, and we don't know when a reviewer would write a spam review.

Despite of many other obstacles of spam detection, what we mentioned above are the core challenges that we think make simple behavioral heuristics insufficient. To capture sophisticated spammers, we consider more clues than simple behaviors.

2.1.2 Contributions

Our first contribution is to propose a heterogeneous graph model with three type of nodes to capture spamming clues. In our opinion, a user's reviews, all the stores (s)he commented on, reviews from other reviewers who have shopping experiences on the same stores are all clues of telling if this user is innocent or not. Therefore, we use a heterogeneous graph to represent relations among reviewers, reviews, and stores, which are three different kinds of nodes in the graph. A reviewer has a link with

a review if (s)he wrote it. A 'review' node links to a 'store' node if it is on the store. A store links to a reviewer through the review on that store. Each node also has a set of features, e.g., a store node has features for its average rating of its reviews, the number of reviews it has, etc. Figure 4 illustrates such a review graph.

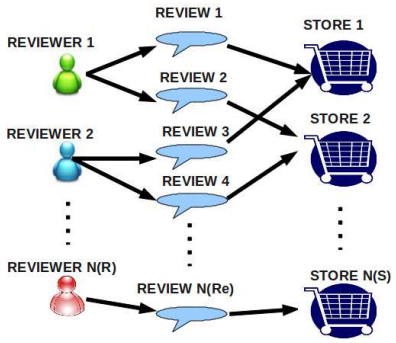


Figure 4. Review Graph

Our second contribution is to introduce three fundamental concepts. The first is the *trustiness* of reviewers: one reviewer becomes trustworthy if (s)he has many honesty reviews. The second is the *reliability* of stores: one store becomes reliable if it has many positive reviews from trustworthy reviewers. The third one is the honesty of reviews: a review is honest if it is agreed by many other honest reviews. These are the inner-relationship between those concepts.

Our third contribution is to develop an effective iterative computation method for the three concepts, based on the proposed graph model.

To the best of our knowledge, we are the first to utilize review graph to explore review spam detection problem and consider more hidden clues than previous solutions.

This chapter is organized as follows: In section 2, we discuss previous work on spam detection and their limitations. In section 3, we formally define the spam detection problem and propose our solution in detail. In section 4, we show our experiment results on real world data. Section 5 is the conclusion.

2.2 Related Work

Paper(40) originates the problem of spam reviews detection. They identified 3 categories of spams: fake reviews (also called untruthful opinions), reviews on brand only and non-reviews. Their spam detection method uses supervised learning. They first extract features on review, reviewer, and product. Then they semi-automatically labeled spam reviews using mainly text similarity and some human effort. Those labeled spam reviews are duplicate or near duplicate reviews, and are treated as positive training examples and the rest are used as negative ones. Based on the features and the training data, a classifier can be constructed to detect spam reviews. Their approach relies heavily on text similarity, therefore it is only good for certain types of spamming activities, i.e., duplicated review spamming.

In (41), the authors proposed another spam review detection algorithm using class association rules. They treated each review as a record associated with a rating class (positive, negative and neutral). A rule mining algorithm is used to produce a list of *unexpectedness* rules. Their study found that the top unexpected rules indeed imply abnormal reviews and reviewers. However, their method does not identify true spammers, but only finds some strange behaviors as unexpected rules.

(56) goes one step further by studying the behaviors of reviewers in order to detect spammers, rather than detecting the spam reviews directly. They also find several features to capture of spamming

behavior, e.g., multiple ratings/reviews on a single product or a group of products, and rating deviation. Each reviewer has some scores on these features. And a linear combination of the scores indicates the suspicious degree of a reviewer. The method is unsupervised which saves a lot of human labeling efforts. However, this work essentially also relies on duplications: multiple reviews from the same reviewer targeting the same item or item group. Thus, it is only suited for a special kind of spamming. Although our work also exams reviewers' behaviors, it is only one small part of the key solution where review graph plays an important role of discovering spamming clues, instead of studying individual behavior of every reviewer as in (56). Moreover, we use different criterion to capture these behaviors to existing work.

(89) developed a spam review detection algorithm based on the ranking of products(in their case, hotels). Their observation is that spam reviews may distort product ranking more than random chosen reviews. Their methods is restricted to the situation where the ranking of products are available, which cannot be applied to general review datasets.

Paper (66) studied a special spamming activities, i.e., group spamming, where a group of spammers are assumed to write spams together on a few target stores. Their focus is also a subset of spamming and not applicable to general cases.

Comparing with all the previous work, ours is the first one to explore how a reviewer's relations with other reviewers and stores reveal clues of spamming. Complementing existing work, our method is able to find more subtle and sophisticated spamming activities, using reinforcements of reviewer's trustiness, review's honesty, and store reliability.

The general idea of reinforcement based on graph link information has been applied in many different scenarios. PageRank (67) and HITS (47) are successful examples in link based ranking using reinforcement. But they are not applicable in our spam detection case. PageRank can discover prestige nodes in a graph. But prestige is not related to spamming activities. We cannot say a prestige node has larger chance to be a spammer or a benign reviewer. HITS has authority and hub nodes influencing each other, which is only conceptually similar to our three types of nodes interaction but with *no* other comparable details. In HITS, authority's weight is the summation of the weights of hubs linked to it. In our case, simply using summation of one score of *trustiness, reliability, or honesty* to infer another would not work. We derived the interaction functions uniquely from spam detection domain properties(see Section 3.1 for detailed discussion). They turned out to be non-linear functions that are better suited for this problem. Moreover, we have three different types of nodes and each node also has attached features, which don't exist in both PageRank or HITS.

Reinforcement is also used in discovering truthful information from multiple conflicting sources (92), and identify fair research paper reviews in an evaluation system (51). Other kinds of link based analysis, e.g., belief propogation, have also been used for detection of other forms of untruthful information, such as online auction fraud (69) or accounting risk (65). Both their problem settings and detail techniques are different from spam detection, therefore they are not applicable to our work.

2.3 Spam Detection Model

In this section, we first introduce the assumptions based on which we build the proposed model. Then we define three factors in the model, namely, reviewers' *trustiness*, reviews' *honesty*, and stores'

reliability. At the end of the section, we discuss the iterative computation framework, which computes the three factors.

2.3.1 Intuitive Assumptions and Observations

We start by exploring possible causes of spamming activities. Grounded on common sense, we first make the following assumptions:

- Spammers are usually for profits, so they have connections to stores that would benefit spammers to promote their prominence or defame other stores (2).
- There are several categories of stores in terms of their quality. High quality stores are excellent in terms of their product qualities and customer services. Low quality stores are either junk ones that sell poor quality products or fake ones that never deliver products.
- Spammers are usually hired by low quality stores. We make this assumption based on a couple of reasons.
  - Low quality stores are the ones that suffer if customers know the truth about them.
  - They are the ones in desperate need to be promoted by customers, which may be the only way to get attention of more customers.
  - They are the ones that are less competitive, but want to defame their competitors in order to get themselves more profits.

  Due to the above reasons, such stores have stronger motivation to hire spammers to write dishonest reviews. On the other hand, stores with a higher reputation, stable consumer population, and good revenue may not hire spammers at all, since they lose much more if they are caught doing so.

Even if good stores really entice spammers to say good things about themselves, it may not be very harmful. Therefore, we assume that less reliable stores are more likely involved in review spammings.

- Harmful spam reviews always try to deviate from the truth. Therefore, they can be either positive reviews about lousy stores, or negative reviews about good stores.
- Not all reviews deviating from mainstream are spams. People may feel differently or have different experiences about the same service.

Based on these assumptions, we can infer that enthusiastic praise about low quality (therefore less reliable) stores and unrealistic complains on high quality (therefore more reliable) stores are more suspicious. Although there may be exceptions in practice, these types of spammers are our main concerns, since we believe that they cover a significant amount of harmful spamming activities. Therefore, we have the following observations:

1. We can judge the honesty of a review given the reliability of the store it was posted to, plus other surrounding reviews' agreement (to be defined later) with the same store.
2. If we have the honesty scores of all the reviews of a reviewer, we can infer his/her trustiness, because one is certainly more trustworthy if one wrote more reviews with high honesty scores.
3. Now we go back to see how to depict a store's reliability. From common sense again, a store is more reliable if it is reviewed by more trustworthy reviewers with positive reviews, and less reliable if it is reviewed by more trustworthy reviewers with negative reviews.

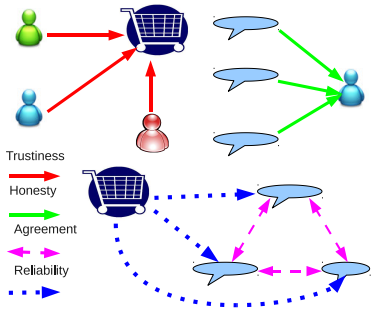


Figure 5. Influences among different types of nodes

Figure 5 shows the above influences among a store's *reliability*, a review's *honesty*, and a reviewer's *trustiness*. They are intertwined together and reinforcing each other. These influences are conceptually like the well-known authority and hub relations (47). Nevertheless, relations in our case are quite different from those in authority-hub analysis. First, a reviewer cannot gain more trust by writing more reviews. Nor could we use the mean of review honesty to capture a reviewer's trustiness. Second, reviews have impact on each other. If a review deviates from most of the others, it could be a clue of spamming.

2.3.2 Basic Definitions

From the above observations, we define variables that quantify credits of reviewers, reviews, and stores.

Definition 2.3.1 (Trustiness of reviewers) *The trustiness of a reviewer r (denoted by* $T(r)$ *) is a score of how much we can trust r. For ease of understanding and computation, we limit the range of* $T(r)$ *to* $(-1, 1)$.

Notation	Definition
$r(i)$	reviewer r's i^{th} review
$r(s)$	reviewer r's review on store s
$n(r)$	the number of r's reviews
H_r	the honesty summation of r's reviews
$rating(re)$	review re's rating
$aid(re)$	review re's author id
$sid(re)$	store id of re's
$t(re)$	review re's posting time
U_s	the set of reviews on store s

TABLE II

FEATURES ASSOCIATED WITH NODES AND THEIR NOTATIONS

Definition 2.3.2 (Honesty of reviews) *The honesty of a review re (denoted by $H(re)$) is a score representing how honest the review is, $H(re) \in (-1, 1)$*

Definition 2.3.3 (Reliability of stores) *The reliability of a store s (denoted by $R(s)$) is a score representing the quality of store s. $R(s) \in (-1, 1)$*

Table II lists node features that are used in our algorithm and their notations.

2.3.3 Reviewers Trustiness

To model the trustiness, let's see how we can tell if a reviewer is trustworthy or not, based on which we can devise several heuristic ideas.

1. A reviewer's trustiness doesn't depends on the number of his/her reviews, but on the summation of their honesty scores.

   For example, a reviewer with ten reviews could be less trustworthy than a reviewer with only one review, if those ten are fake and the one is honest. But if two reviewers have similar average review honesty scores, one is more trustworthy than the other if (s)he has more honest reviews.

2. We trust a reviewer if his/her reviews tell the truth. We don't trust a reviewer if his/her reviews are suspicious.

   This means a reviewer's trustiness score should be positive if he/she writes more reviews with positive honesty scores. Vice versa, the score should be negative if (s)he has many reviews with negative honesty scores.

3. For a given reviewer, his/her trustiness does not grow/drop linearly with the number of high/low honesty reviews that (s)he wrote. It grows/drops faster when the number of such reviews is smaller, and slows down when the number is larger.

   For example, when a reviewer has already written 100 honest reviews hence becomes highly trustworthy, his/her trustiness does not improve that much by writing one more honest review. However, if we have a reviewer with only 2 reviews, his/her trustiness will increase significantly as the third high honesty review appears.

Here we define a reviewer r's trustiness to be dependent on the summation of the honesty scores of all r's reviews,

$$H_r = \sum_{i=1}^{n(r)} H(r(i)) \tag{2.1}$$

where $n(r)$ is the number of reviews from r.

The proposed trustiness score of r should be a function T satisfying the above intuitions that are formally represented by the following relations.

$$T(i) < T(j),\ if\ H_i < H_j \tag{2.2}$$

$$T(r) < 0,\ if\ H_r < 0,\ T(r > 0)\ if\ H_r > 0 \tag{2.3}$$

$$\frac{dT(r)}{dH_r} = T(r)(K - T(r)). \tag{2.4}$$

Relation (2) represents the first heuristic: One reviewer is more trustworthy than another if one has larger honesty score. Relation (3) depicts the second heuristic: We don't trust a reviewer whose reviews tend to be dishonest, but we trust a reviewer whose reviews tend to be honest. Relation (4) represents the third heuristic that is similar to the population growth model (70): The growing (or dropping) speed of trustiness is the product of current trustiness level and the room of improvement. Suppose K is the upper bound of trustiness score, $K - T(r)$ is how much more trustiness one can get by writing more

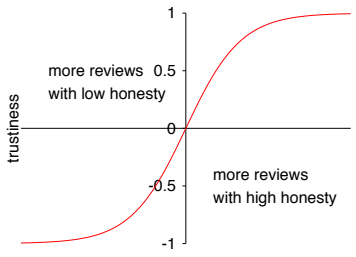


Figure 6. Relation between *trustiness* and *honesty*

honest reviews. It becomes smaller as $T(r)$ grows. Solving these relations gives us the general form of trustiness function

$$T(r) = \frac{K}{1 + e^{-KH_r}} \tag{5}$$

Notice this general form is in range $(0, K)$. As we mentioned before, the upper bound of trustiness score is $+1$ in our case, and its range should be in $(-1, 1)$ to have more practical meanings.

Therefore, we re-scale the trustiness *T* of a reviewer r as

$$T(r) = \frac{2}{1 + e^{-H_r}} - 1 \tag{6}$$

The general shape of trustiness function is the well-known logistic curve and is shown in Figure 6.

In reality, reviews from the same reviewer can reflect different degrees of facts, due to many reasons, e.g., subjective bias. A normal reviewer p may post an unrealistic review i with $H(p(i)) < 0$, while a spammer q may have honest review j with $H(q(j)) > 0$. However, it is still possible to distinguish

malicious reviewers from benign ones, since the trustiness score is controlled by the summation of a reviewer's review honesty, i.e., the general trend of a user's reviews.

To calculate $T(r)$, we need to know the honesty values of r's reviews, which we will define in the next subsection.

2.3.4 Review Honesty

How do we interpret a review? When we shop online and read a review, we usually keep two factors in our mind. The first one is the store we are looking at. If the store is a good one such as *Apple.com*, we tend to trust the positive reviews on it. If we are reading reviews about a store we have never heard of, or we knew it was a bad one, we tend to doubt the high rating reviews posted on it. The second factor is the surrounding reviews, which are other reviews about the same store within a certain time window Δt, e.g., 3 months before and after the posting time of the target review. We are likely to trust the mainstream opinions held by most surrounding reviews, rather than abnormal opinions.

Based on these observations, we model a review's honesty by two factors.

1. The *reliability* of the store it comments.
2. The *agreement* between this review and other reviews about the same store within a given time window.

We will discuss *reliability* in the next subsection. We study *agreement* first by introducing the surrounding set.

Definition 2.3.4 (Surrounding Set) *The surrounding set of review re is the set of re's surrounding reviews.*

$$S_{re} = \{i \mid sid(i) = sid(re),\ |t(i) - t(re)| < \Delta t\}$$

$\forall i, j \in S_{re}$ agree with each other when their opinions about the same aspects of the store are close. However, opinion mining is too costly to let us assess opinion of a reviewer (36)(71). Fortunately, in addition to the review text, reviews usually have rating information about a store. Even if two 5-star ratings may mean different aspects of a store, such as customer service and delivery, they are correlated to each other. Therefore, we make some assumptions here.

- A review's rating about a store reflect its opinion.
- Two reviews with similar rating scores about the same store have similar opinions about the store.

From the assumptions, $\forall i, j \in S_{re}$ agree with each other if

$$|rating(i) - rating(j)| < \delta \tag{7}$$

where δ is a given bound (we use 1 in a 5-star rating system in this chapter).

Thus, we can partition the surrounding set S_{re} as

$$S_{re} = S_{re,a} \cup S_{re,d} \tag{8}$$

$$S_{re,a} = \{i \mid\ |rating(i) - rating(re)| < \delta\} \tag{9}$$

$$S_{re,d} = S_{re} \setminus S_{re,a} \tag{10}$$

We also put the reviewers' trustiness scores into consideration. One review should be good even if it does not agree with any surrounding reviews when it is written by a trustworthy reviewer while the surrounding reviews are posted by untrustworthy reviewers. Similarly, one review may be bad even if it agrees with surrounding reviews, since they may all come from spammers. Therefore, we define the agreement score of review re within time window Δt as

$$A(re, \Delta t) = \sum_{i \in S_{re,a}} T(aid(i)) - \sum_{j \in S_{re,d}} T(aid(j)) \tag{11}$$

Notice that trustiness score T could be either positive or negative. This equation means that if one review agrees with other reviews by trustworthy reviewers, its agreement score becomes higher. On the other hand, if it agrees with untrustworthy reviewers, its score becomes lower. This equation also promotes a benign user's review agreement score when it is surrounded by spam reviews that it does not agree with. Because if spammers' trustiness scores are negative, a benign review's agreement score gets promoted by subtracting the negative numbers.

$A(re, \Delta t)$ can be positive or negative. Here we normalize it to $(-1, 1)$ to make later computation easier.

$$A_n(re, \Delta t) = \frac{2}{1 + e^{-A(re, \Delta t)}} - 1 \tag{12}$$

Review re's *honesty* $H(re)$ is defined as follows.

$$H(re) = |R(sid(re))| A_n(re, \Delta t) \tag{13}$$

where $R(sid(re))$ is the *reliability* of store $sid(re)$, which we will define later. But what we clarify here first is that, by definition, $R(sid(re))$ can be positive (for reliable stores) or negative (for poor quality stores). We take its absolute value as an amplifier of $A_n(re, \Delta t)$. This is consistent with previous discussions. If a store's $|R(sid(re))|$ is large, it is either quite good or quite lousy. A review on this store should have high honesty score if it agrees with many other honest reviews. If $|R(sid(re))|$ is small, the store's reliability is difficult to tell. And a review's honesty score gets diminished a little bit by that fact.

2.3.5 Store Reliability

When we define reliability, we have similar intuitions as when we define *trustiness*. A store is more reliable if it has more trustworthy reviewers saying good things about it, while it is more unreliable if more trustworthy reviews complains about it. The growing/dropping trend of reliability would also be a logistic curve so as to be consistent with our common sense. Therefore, we define the reliability R of store s as

$$R(s) = \frac{2}{1 + e^{-S_T}} - 1 \tag{14}$$

where

$$S_T = \sum_{r \in U_s, T(r) > 0} T(r)(rating(r(s)) - avgRating) \tag{15}$$

and $avgRating$ is the median value of the entire rating system, e.g., 3-star in a 5-star rating range.

Therefore, a store's reliability depends on all *trustable* reviewers who post reviews on it, and their ratings. When considering reliability, we only consider reviewers with positive trustiness value because their rating really reflect the store's quality. In contrast, whatever a less trustworthy reviewer says about

a store, it is less trustable. For example, we don't know the real intention to rate a store as a good one or bad one for a review, if it comes from a potential spammer.

2.3.6 Iterative Computation Framework

Integrating the pieces of information of the review graph together, we have an iterative computation framework to compute reliability, trustiness, and honesty, by exploring the inner dependencies among themselves. Here we illustrate the whole algorithm.

In every round, to compute trustiness T for every reviewer, we only need to access each review once and get its honesty value. Therefore, this step is linear to the number of reviews N_{re}. For a similar reason, reliability R computation is linear to the total number of reviewers N_r. We will show in Section 4.1 that in our case, the computation of honesty H for all reviews is also linear to the number of reviews N_{re}, from a probabilistic point of view. Therefore, the overall time complexity of our framework is

$$O(k(N_r + N_{re}))$$

where k is the number of iterations. k is small (usually 4 or 5) as the algorithm converges quite fast (see Section 4.2)

2.4 Evaluation

In this section, we discuss the details of the real world dataset that we use for our experiments, the computational performance of our model, and human evaluation results. All the experiments are run on a Intel(R) Pentium(R)-D machine with 2.20 GHz CPU and 3GB of physical memory. The whole system is implemented in Java.

2.4.1 Data Set and Data Features

We use the review data from *www.resellerratings.com* for our experiments [1]. It is one of the largest hosts of online store reviews. The website provides a unique url for every reviewer's profile, containing meta data such as the reviewer' id, join date, whether a 'top reviewer' or not, all his/her reviews and ratings with posting times about stores, and links to those stores. At every store's page, the website has information about its average rating score, all reviewers with their reviews, and other features like 6-month rating, and consumer protection plan enrolled or not.

The data we crawled is a snapshot of complete information from the website on Oct. 6th, 2010. We clean the data by removing users and stores with no review. After that, we have 343603 reviewers who wrote 408470 reviews on 14561 stores in total.

As we discussed before, skilled spammers can manipulate their review text and other behaviors, e.g., posting time, to make themselves look normal. Therefore, in our model we only use essential data such as reviewers' ids and their reviews with ratings and their relations with stores to build the review graph. In fact, other information from the website may be ad-hoc and a black-box for us. For example, 'top reviewers' defined by the website are not the most trustable reviewers, but the ones with most reviews [2]. And the "average rating" information displayed for a store is sometimes missing. Therefore, we re-calculate the average rating of each store based on all its reviews' ratings for evaluation purpose(see Section 4.4).

---

[1] Thanks to Keith Nowicki for data collection

[2] http://www.resellerratings.com/topreviewersalltime

2.4.2 Computational Performance

Figure 7 shows the convergence of the iterative computation. Let $\mathscr{T}i$ and $\mathscr{T}i+1$ be the vectors of *trustiness* scores of iteration i and the i+1. The change

$$c_i = 1 - cos(\mathscr{T}i, \mathscr{T}i+1)$$

converges quickly after a few iterations.

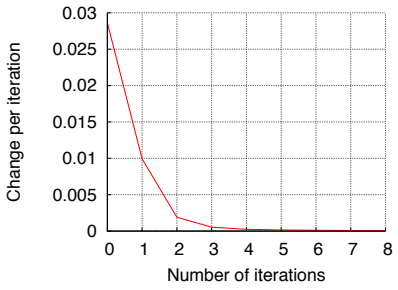


Figure 7. Convergence

Now we analyze the complexity when computing review *honesty*. As we defined in the previous section, the honesty of a review is determined by the reliability of the store it comments on and the agreement between itself and other reviews of its surrounding set. To access the reliability score of a given store is only an $O(1)$ operation. However, it seems to cost $O(m^2)$ to compute agreement of a review re, where m is the number of reviews of store $sid(re)$. Fortunately, the review number

distribution follows *power-law* (8) in the review graph. It means that only a few stores have many reviews, while most stores have only a few reviews.

Figure 8 illustrates the review number k's probability distribution over stores. It is a double-log plot of review distributions. Fitting it with a straight line, we have

$$P(k = m) = \alpha m^{-1.43}$$

Therefore, total operations on computing agreement in one iteration is

$$\begin{aligned} Op(agreement) &= \sum_m P(k = m)m^2 = \sum_m \alpha m^{-1.43} m^2 \\ &= \sum_m \alpha m^{0.57} < \sum_m \alpha m = O(N_{re}) \end{aligned}$$

that is linear in the total number of reviews.

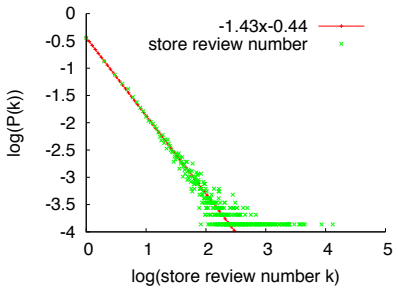


Figure 8. Review distribution

2.4.3 Spammer Detection Results Evaluation

2.4.3.1 Evaluation Criteria

Review spam detection's unique challenge lies in its lack of ground truth, i.e., the label indicating which review is a spam or which reviewer is suspicious does not exist naturally in the review data. To obtain such labels, human judges are needed to read the reviews, search the internet regarding the truth of the reviews, and use their intuitions to make the judgement, which requires significant amount of maneuver. Thus, it costs too much time for human judges to go over every reviews and label them. Instead, we let our algorithms identified highly suspicious spammers candidates first, then we recruit human judges to evaluate how many of them are really suspicious, in such way we exam our algorithms' performance.

Similar evaluation approach has been used in previous review spam detection research (56)(40), therefore this is a well-established way of performance measure. The details of our evaluation strategy is as follows.

IR-based Evaluation Strategy. The goal of spammer detection method is to identify suspicious reviewer candidates for further investigation. Thus, it is similar to an information retrieval task, which tries to present the users the most interesting items first. Therefore we borrow evaluation standards from information retrieval to show the performance of our spammer detection algorithm. There are two such standard measures: precision and recall. For our task, precision is defined as the fraction of spammers from all candidates retrieved by the algorithm; recall is the percentage of retrieved spammers among all *real* spammers. Since we have no ground truth about *real* spammers, we simply use the precision as one evaluation measure.

Human Judgments Consistency Criterion. Since there is no "spammer" label in any kind of review data, human evaluation is necessary to judge if a target is untrustworthy. A spammer detection algorithm is reliable, if different human evaluators agree with each other about their judgments on the same set of results. Therefore, we use human judgment consistency as another important criterion.

Our human evaluators are 3 computer science major graduate students who also have extensive online shopping experiences. They work independently on spammer identification.

2.4.3.2 Human Evaluation Process

Judging suspicious spammers is a complicated task for human and often involves intuition and looking for additional information, especially when we target at more subtle spamming activities. To decide if a candidate is a spammer requires human judges not only to read his/her reviews and ratings, but also to collect evidences from relations with other reviewers, stores, and even the Internet.

To standardize this complex process, our human evaluators agree upon three conditions and put them together as our evidence to claim that a reviewer is a potential spammer.

- *A reviewer is suspicious if (s)he has a significant number of reviews giving opposite opinions to others' reviews about the same stores.*

  For example, if a reviewer gives high ratings to all the stores he has reviewed, while other reviewers usually rate these stores low, the reviewer is problematic.

- *A reviewer is suspicious if (s)he has a significant number of reviews giving opposite opinions about some stores as compared to the ratings from the Better Business Bureaus (BBB)* [1]

  For example, if a reviewer gives high ratings to all the stores (s)he reviewed, but BBB gives them Fs (the rating range is from A to F), the reviewer is clearly problematic.

- *A reviewer is suspicious if (s)he has a significant number of reviews saying opposite opinions about some stores as compared to evidences presented by general web search results.*

  For example, if a reviewer gives high ratings to all stores he reviewed, but Google search results about these stores often contain information of them being scam or having fake reviews, this reviewer is problematic.

Although every single condition may not be convincing enough to prove spamming activities, all of them together can be a confident claim of spamming activities. We evaluate the top 100 suspicious reviewers identified by our model. No existing work focused on such a large scale and subtle individual cases. Human evaluators gave their independent judgments based on various information from reseller-ratings.com, business honesty information from BBB, and search results from Google, and by reading reviews, according to above conditions.

2.4.3.3 Precision and Consistency

In our evaluation, if more than one evaluators regard a reviewer as a spammer, we label it a suspicious spammer. With limited evidence from the Internet, our evaluators identify 49 out of 100 suspicious

---

[1] *Better Business Bureau is a well-known corporation that endeavors to a fair and effective marketplace. It collects reports about how reliable a business is, alerts the* general *public about business/consumer scams, and enforce the mutual trustiness between consumers and companies*

candidates to be spammers. The precision is 49%. Although our precision is not very high compared to the previous work, we are dealing with much more subtle and complex cases (not simple duplications), which existing studies could not handle. Besides, our precision is meaningful, since human evaluators agree with each other on their judgments.

Table III shows the agreement of human judges. For example, Evaluator 1 identified 49 suspicious reviewers, out of which 33 were recognized by Evaluator 2 and 37 were caught by Evaluator 3. To explore their agreement, we use *Fleiss' kappa* (24), which is an inter-evaluator agreement measure for any number of evaluators. The kappa among 3 evaluators is 60.3%, which is almost substantial agreement (50). It is also important to note that those reviewers who were not identified as spammers are not necessarily innocent. It is simply because our human judges have not found enough strong evidences to conclude that they are spammers.

	Evaluator 1	Evaluator 2	Evaluator 3
Evaluator 1	**49**	33	37
Evaluator 2	-	**35**	23
Evaluator 3	-	-	**40**

TABLE III

HUMAN EVALUATION RESULT

2.4.3.4 Compare with Baseline

Since our work is the first one utilizing review graph and targeting subtle spamming activities, there is no existing work that is comparable. Besides, previous studies are mainly focusing on different forms of duplicate reviews, in order to catch spammers. Given the differences between our work and previous studies, we want to demonstrate that suspicious reviewers found by our method can hardly be identified by available techniques.

We choose to compare with the approach in Lim et al. (56), because it is the state-of-art of behavior based spammer detection techniques. They use some types of duplicate reviews as a strong evidences of spamming. They first look for candidates who have multiple reviews about one target (in our case store), and then compute spamming scores to capture spammer from candidates.

In our top 100 suspicious candidate list, only 3 candidates can be found based on their criterion. Moreover, only 1 out of these 3 candidates is finally labeled as suspicious spammer by our human evaluators. This result means that our work detects different types of spamming activities from existing researches. They can seldom find the spammer types that we are able to find. By no means do we claim that the existing methods are not useful. Instead, our method aims to find those that cannot be found by previous methods.

2.4.3.5 Suspicious Spammer Case Study

Here we take a close look at reviewers ranked highly suspicious by our algorithm. We pick two candidates: one often promotes stores while the other demotes stores.

The first case is reviewer *howcome* [1], whose reviews are mostly positive. When we examined these reviews, we found many of them are problematic, such as highly rated reviews for UBid, OnRebate, ISquared Inc, Batteries.com, and BigCrazyStore.com. For example, UBid is widely complained at different review websites like ConsumerAffairs, ePinions, CrimesOfPersuasion, and ResellerRatings. OnRebate is *sued* for failing to pay rebates to customers. ISquared Inc is rated as D by BBB. Batteries.com is generally low rated at ResellerRatings. BigCrazyStore has very few records about its quality on the Internet. And this reviewer's review is the only one about BigCrazyStore on ResellerRatings. All these evidences lead us to conclude that this reviewer is suspicious.

The second case is *shibbyjk* [2]. All reviews of this reviewer are complains. And all stores being complained are high quality companies according to BBB, e.g., 1SaleADay(A), StarMicro(B^+), 3B Tech(B), and Accstation(A^-). From these unfair ratings, one may argue that this reviewer may be still normal and just picky. However, after reading all his/her negative reviews, we found that all complains are about *transaction problems*. It is fishy because the chance of all these high standard companies having transaction problems and credit card frauds with this particular customer is very low. Besides, *transaction problem* is a good excuse to make unrealistic complains, since the truth is hard to be verified.

---

[1] http://www.resellerratings.com/profile.pl?user=57665

[2] http://www.resellerratings.com/profile.pl?user=565595

2.4.4 Store Reliability Evaluation

Our algorithm can also give every store a reliability score. In this subsection, we pick the top 10 and the bottom 10 stores to show their ratings on ResellerRatings.com and BBB. In this way, we demonstrate the accuracy of reliability scores our model can offer.

Store Name	Reselleratings Rating	BBB Rating
TigerDirect	7.44	A
SuperMediaStore	9.27	A^+
OneCall	9.33	A^+
Newegg	9.77	A^+
Mwave	9.18	B^-
LA Police Gear	9.11	A^-
iBuyPower	8.33	B^-
FrozenCPU	9.44	A^+
eWiz	9.08	C
eForcity	8.55	A^-

TABLE IV

TOP 10 RELIABLE STORES

Tables IV and V list top and bottom stores identified by our algorithm. (We randomly choose a store if multiple stores have the same reliability score.) As we can see, our algorithm generally gives high reliability scores to quality stores, and low scores to lousy ones. However, there are a few exceptions,

Store Name	Reselleratings Rating	BBB Rating
86^{th} Street Photo	0.30	F
Best Price Cameras	1.43	F
Dealer Cost Car Audio	1.23	F
USA Photo Nation	0.20	F
Camera Addict	0.59	F
CCI Camera City	0.44	A^+
OC System	3.00	F
Shop Digital Direct	0.35	F
Camera Giant	0.21	F
Infiniti Photo	0.28	F

TABLE V

BOTTOM 10 RELIABLE STORES

where the BBB and reslleratings are not consistent: one good and one bad. For example, CCI Camera City is an interesting case. Its BBB rating is quite high. However, when we simply put this company's name into Google, we found 6 out of 10 results at first page are customer complaints against it. This phenomenon means that our algorithm works and no source could provide 100% accurate information about a company's quality, e.g., BBB also has blindspots. Another inconsistent case is eWiz, but we have not enough evidence to tell which rating would be close to the truth.

2.4.5 Review Honesty

As the third type of nodes in the review graph, every review also has a honesty score. However, unlike a reviewer or a store, it is even harder to evaluate a single review. We can have human evalu-

ators for reviewer evaluation, and we have Internet information for store evaluation. But we have no convincing evidence sources to judge every single review to tell whether a few sentences are true or not. Besides, unlike trustiness or reliability, it may not be that informative to study single review's honesty. Therefore, we omit the details in this part. It is also shown by previous studies that individual reviews are very hard to judge [6, 7, 11]

CHAPTER 3

INFLUENCE AND SIMILARITY ON HETEROGENEOUS NETWORKS

3.1 Introduction

In social network research, the studies on social influence maximization and entity relevance are two important and orthogonal tasks. On homogeneous network, social influence maximization research tries to identify an initial set that maximizes information spreading, while relevance studies focus on designing meaningful ways to quantify entities' similarities. When heterogeneous networks are becoming ubiquitous and entities of different types are related to each other, we observe the possibility of merging the two directions together to improve the performance for both of them. On the one hand, the influences among one type of nodes could be defined more accurately with the consideration of relevance of the other type of nodes because of their inter-connections. On the other hand, the computation for relevance relations of the other type of nodes could also be more informative with the input of the influences of the former type of nodes.

Therefore, we introduce a framework that computes social influence for one type of nodes in a heterogeneous network, and simultaneously measures relevance of the other type of nodes. Firstly, we decouple the target heterogeneous network (or we call it *Influence Relevance (IR)* network) into three different parts: influence graph (*IG*), relevance graph (*RG*) and information tunnels (*IT*) between *IG* and *RG*. Then through *IT*, we exchange the influence scores and the relevance scores to calculate more precise relevance and influence scores in order to improve both of their qualities. Our mutually

reinforced algorithm stops when these scores converge. The experiment results on real world data show that our framework enables influence maximization framework to identify more influential seeds in *IG* and relevance measures to produces more meaningful relevance scores in *RG* simultaneously.

Social network is a popular topic in research community. People have proposed many novel directions of analysis on social network to explore its properties of many different aspects, e.g., structure and information spreading flow. Two prominent techniques of such analysis are *social influence maximization* and *entity relevance analysis*. Given a user population and their relations in an information cascade scenario, social influence maximization is centered on constructing a meaningful influence network, identifying a set of most influential nodes in the network, and maximizing the spread of influence through such a social network. It enriches our knowledge about dynamic information handling behaviors of nodes in social networks and helps us design applications such as virtual marketing, recommendations, and information retrieval (44)(80). Relevance analysis, as another flourishing research direction, is proposing methods for measuring nodes similarities, based on the network structure and node features. Knowing relevance relations among nodes gives us tools to perform many fundamental tasks such as clustering or classification, as well as useful applications such as information retrieval, ranking, or recommendation(75)(94). Although influence and relevance analysis could provide tools for similar applications, they are often considered as orthogonal techniques that are studied separately, besides they are often applied to homogeneous networks. When the heterogeneous network is becoming ubiquitous, we notice that it is not only necessary but also beneficial to combine the two techniques into one framework because we can use information from one side to calibrate the other side.

3.1.1 Improving Influence Maximization

Given a set of users and their activation relations, the ultimate goal of social influence maximization research is to identify limited number of influential people as seeds from which the spreading of information entity would be maximized. An influence network of the users is first constructed according to such relations. Algorithms that depict different information cascading rules are developed on the network to explain the real cascading phenomena.

Previously, majority of such research assume that the cascading network is given and so are activation probabilities among nodes (44)(16) which is either a fix number for every node or weighted value of a node's degree. Their major focus is to design different cascading algorithms, e.g., independent cascade, linear threshold, and compare their performance on the cascading coverage, i.e., the number of activated people in the given network.

There are two important phenomena that are constantly overlooked by previous research, to the best of our knowledge. The first one is that they seldom demonstrate the **seed qualities** that are selected by their algorithms. One possible reason is that majority of previous reasearch used virtual marketing data where it was hard to verify who were indeed influential to other users in a dynamic virtual marketing environment. The second overlooked phenomenon is the definition of activation probability. Under which circumstance would a user activate another, i.e., pass over the information, is an important factor to the whole cascading process. However, we have only found 1 publication (29) that explicitly studies how to calculate such probabilities, and their work is in a totally different problem setting.

Our approach addresses the two overlooked factors by introducing similarity measures. In a heterogeneous network, the activation probability between two nodes of the same type is often related to the

other type of nodes they connect to. Considering such connections would offer us a more precise way of activation probability model and a better influence maximization result.

3.1.2 Improving Similarity Measure

The research community has been proposed numerous similarity measures, symmetrical or asymmetrical, for nodes in social networks, which consider node features (17), link features (28), and other semantic features (26). Also, the similarity measures have been defined over either homogeneous or heterogeneous networks among same type of nodes or different types of nodes. A simple clue for designing meaningful similarity measures is to customize the definition and consider more information according to application scenario.

In our case, due to the heterogeneous nature of the network, we should put the influence of one type of nodes into the formulation when computing the similarity of the other type of nodes. This leads to an asymmetric similarity formulation. This approach has never been studied before. As we analyzed in the above subsection, introducing similarity to influence maximization in a heterogeneous network could be beneficial to many key aspects on influence maximization. Further more, similarity measure gets more customized information from influence maximization side, which should potentially be beneficial too. Fortunately, our experiment results confirm the **mutual beneficial relation**. Below is a quick case study on this point.

3.1.3 Brief Case Study

For example, in a *bi-typed* academic social network (Figure 22(b)), there are paper-paper citation relations, author-paper relations, and author-author reference relations. Let us first concentrate on paper citation relations. Among all citations of a specific paper, some are more relevant to it than others, since

they may be the work that inspire the paper while others may only get generally mentioned. Authors' influences should be part of the computation because when compiling citations, one tends to select the work from authors who impact oneself more. Now let us pay attention to author reference relations. In influence maximization computation among authors, whether an author can "activate" another author depends on not only their connections, but also their papers' relevance. Even if author A cites equal number of papers from B and C, A is easier to be activated by B than C if B's papers are more relevant to A's.

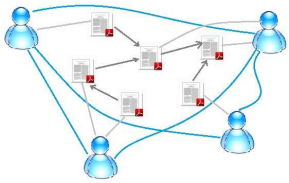


Figure 9. Academic Social Network

On influence maximization part, Tables VI is a quick case studies on the benefits of such calibrations. It lists top seeds selected by our method (**I**nfluence and **R**elevance (IR) model) and the classic **I**ndependent **C**ascade (IC) method. As we can see, the seeds selected without such calibrations (second column), despite they are promising researchers, are not as influential as those are in the first column.

On the relevance analysis part, Table VII gives a comparison on most relevant papers to the *FP-tree growth* algorithm, which are identified by IR model and state-of-the-art similarity measure PRank (94).

IR	Original
Jon Kleinberg	Xuanhui Wang
Philip S. Yu	Qiaozhu Mei
Jennifer Widom	Li Chen
Michael Stonebraker	Daniel Kifer
John A. Stankovic	Jimeng Sun

TABLE VI

TOP SEEDS IN DATABASE & DATA MINING FIELDS

With the help from influence analysis, IR identifies top relevant papers are truly important references. Without such help, PRank's selection reveals more randomness.

Thus, relevance and influence computation can benefit each other, which motivates us to study how to effectively combine them together in one framework. Our technique adopts reinforcement scheme on top of heterogeneous network de-coupling. To be more specific, we first define a special bi-typed heterogeneous network as Influence Relevance (IR) network; We then de-couple its different types of nodes into two homogeneous networks based on their relations, and we do the maximization of social influence spreading on Influence network and expectation of relevance measures on Influence network. However, the two networks are not totally separated. There is a *latent tunnel* connecting them for the sake of delivering information back and forth to improve the performance influence and similarity analysis.

We summarize our major contributions as follows.

paper	Mining frequent patterns without candidate generation, J. Han, et.al, SIGMOD'00
IR top Relevant citations	Fast Algorithms for Mining Association Rules in Large Databases, R. Agrawal, et.al, VLDB'94
	An Efficient Algorithm for Mining Association Rules in Large Databases, A. Savasere, et.al, VLDB'95
	A tree projection algorithm for generation of frequent item sets, R.C. Agarwal, et.al, JPDC'00
PRank top relevant citations	A tree projection algorithm for generation of frequent item sets, R.C. Agarwal, et.al, JPDC'00
	Scalable Techniques for Mining Causal Structures, C. Silverstein, et.al, VLDB'98
	Efficient mining of emerging patterns: discovering trends and differences, G. Dong, et.al, KDD'99

TABLE VII

MOST RELEVANT REFERENCES OF THE FP-TREE GROWTH PAPER PICKED BY IR AND PRANK (PAPERS MARKED IN BOLD FONT ARE CONSIDERED AS ESSENTIALLY RELEVANT ONES)

- We propose a new angle of treating influence maximization and relevance computation together. To our best knowledge, our work is the first to explicitly explore how to make use of both techniques together to analyze a heterogeneous network in a more comprehensive way.
- We study the mutual improvements of influence maximization and relevance computation on each other. An iterative algorithm with optimization on decoupled heterogeneous networks is tailored for this reinforcement relationship.
- We demonstrate its effectiveness through real world social network analysis. Our method outperforms state of the art in influence maximization and similarity computation when they are performed separately.

This chapter is organized as follows. In the next section, we formally describe our framework in details. In section 3, we show our experiment results on real world social network data. In section 4, we discuss related work. Section 5 is the conclusion.

3.2 Influence-Relevance Computation Framework

3.2.1 IR network and its de-coupling

IR network is a special type of heterogeneous network with edge features of different practical meanings for different edge types (Figure 22(b) is such an example). We have observed that it is generic enough to capture important relations for different types of nodes and explore the hidden reinforcement between influence and relevance. Firstly, we formally define ubiquitously existed **I**nfluence **R**elevance network, or IR network. We then explain necessary concepts related to our model in the next subsection.

Definition 3.2.1 *(Influence Relevance Network) An IR network is a directed heterogeneous network* $\mathcal{G}(\mathcal{V}, \mathcal{E}, \mathcal{F})$ *of two different types of nodes, four types of edges with associated edge features. For ease of presentation, let* $\mathcal{V}_\mathcal{X}$ *be the set of nodes we want to study influence on and* $\mathcal{V}_\mathcal{Y}$ *the set of the type of nodes for relevance research, where* $\mathcal{V} = \mathcal{V}_\mathcal{X} \cup \mathcal{V}_\mathcal{Y}$*. There are four types of edges* E_{XX}*,* E_{XY}*,* E_{YX}*,* E_{YY} *connecting different types of nodes, and* $\mathcal{E} = E_{XX} \cup E_{XY} \cup E_{YX} \cup E_{YY}$*.* $\mathcal{F}$ *is a feature vector associated with different types of edges.* $\mathcal{F} = \mathcal{F}_\mathcal{X} \cup \mathcal{F}_\mathcal{Y}$*.* $\mathcal{F}_\mathcal{X} = \{f_X | \forall e_X \in E_{XX}\}$ *is a vector of variables, each one of which describes the influence scores between two nodes of an edge* e_X*. Similarly,* $\mathcal{F}_\mathcal{Y} = \{f_Y | \forall e_Y \in E_{YY}\}$ *is another vector of variables for relevance scores on the other type of nodes.*

Given an IR network, what worthies notice is that it is **application dependent** on the categorization of $\mathcal{V}_\mathcal{X}$ and $\mathcal{V}_\mathcal{Y}$. On the abstract level, the goal of IR modeling is to maximize the spreading of social influence on nodes $\mathcal{V}_\mathcal{X}$ and simultaneously compute the relevance of nodes $\mathcal{V}_\mathcal{Y}$, so that the results of the two tasks reinforce each other. However, before applying the model, one should fix the one type of nodes as type $\mathcal{V}_\mathcal{X}$ and another type of nodes as type $\mathcal{V}_\mathcal{Y}$, so that the categorization is meaningful to the specific application.

To fulfill our goal to obtain better results for both tasks by calibrating each other, we propose our framework in three steps. What we are introducing now is the first step, IR network decoupling. As majority of previous research on either influence or relevance is on *homogeneous* networks, we want to first decouple IR network into two homogeneous ones with information tunnels on their edges.

Definition 3.2.2 *(IR network decoupling) IR network decoupling is a mapping* $\mathcal{L} : \mathcal{G}(\mathcal{V}, \mathcal{E}, \mathcal{F}) \rightarrow G_X(V_X, E_X, F_X) \times G_Y(V_Y\ , E_Y, F_Y) \times G_{IT}(V_X \cup V_Y, E_{XY} \cup E_{YX})$*, In the mapping, we have* $V_X = \mathcal{V}_\mathcal{X}$*,* $V_Y = \mathcal{V}_\mathcal{Y}$*,* $E_X = \mathcal{E}_{\mathcal{X}\mathcal{X}}$*,* $E_Y = \mathcal{E}_{\mathcal{Y}\mathcal{Y}}$*,* $F_X = \mathcal{F}_\mathcal{X}$*,* $F_Y = \mathcal{F}_\mathcal{Y}$*.* $E_{XY} = \mathcal{E}_{\mathcal{X}\mathcal{Y}}$ *and* $E_{YX} = \mathcal{E}_{\mathcal{Y}\mathcal{X}}$*.*

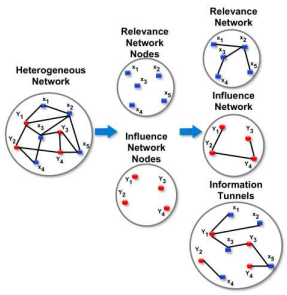


Figure 10. Edge Transformation in IR Decoupling

As seen in the above definition, in the decoupling process, we first preserve the nodes and edge structures for both influence network and relevance network. That is to separate the IR network into two by structurally removing edges $\mathcal{E}_{\mathcal{XY}}$ and $\mathcal{E}_{\mathcal{YX}}$. However, these edges are actually preserved in the $G_{IT}(V_X \cup V_Y, E_{XY} \cup E_{YX})$. We call it information tunnel, since the relevance and influence information can transit through these connections. One should be aware that $G_{IT}(V_X \cup V_Y, E_{XY} \cup E_{YX})$ is a bipartite graph, which represents the connection between V_X and V_Y. Thus its edges have no weights. Moreover, E_{XY} and E_{YX} are complimentary to each other. If an edge $a_X b_Y$ belongs to E_{XY}, $b_Y a_X$ is in E_{YX}. For example, in a paper-author network, that a paper is "written" by an author actually is the same as the author "writes" the paper. Figure 10 illustrates such decoupling process. In the following subsections, we will model the information passing in details in coordination with the influence relevance reinforcement.

3.2.2 Maximizing Influence Spreading on Influence Network

Similar to the state-of-the-art research on social influence maximization, our task on the Influence Network is to identify k seed nodes such that the spreading of information can be maximized. The

influence of z_X on w_X is often represented by the probability that z_X *activates* w_X, in another word, information is passed from z_X to w_X. Here we follow the classic independent cascade (IC) model to simulate the information diffusion. However, instead of having the activation probabilities on every pair of nodes are the same value drawn from uniform distribution, we come up with a more fine-grained activation probability definition. Therefore, our design is the same diffusion process with IC model with finer-grain direct neighbor activation probability.

Let $h(u_X, v_X)$ denote the probability that u_X can activate v_X. Figure 11 demonstrates how we define $h(u_X, v_Y)$ step by step, where each step is explained as follows.

- When two nodes u_X and v_X are neighbors (Figure 11(a)) in the Influence network $G_X(V_X, E_X, F_X)$ (where nodes are circles), the activation probability from u_X to v_X is obtained by considering the relevance scores of their connected nodes in the relevance network (where nodes are squares). We suppose the relevance of i_Y and j_Y is $g(i_Y, j_Y)$, the relevance of k_Y and r_Y is $g(k_Y, r_Y)$, and the relevance of k_Y and l_Y is $g(k_Y, l_Y)$. We consider u_X can influence v_X through $i_Y j_Y$, $k_Y r_Y$ and $k_Y l_Y$ independently. Thus $h_1(u_X, v_X) = 1 - (1 - g(i_Y, j_Y))(1 - g(k_Y, r_Y))(1 - g(k_Y, l_Y))$. More generally, if $u_X v_X \in E_X$, we define

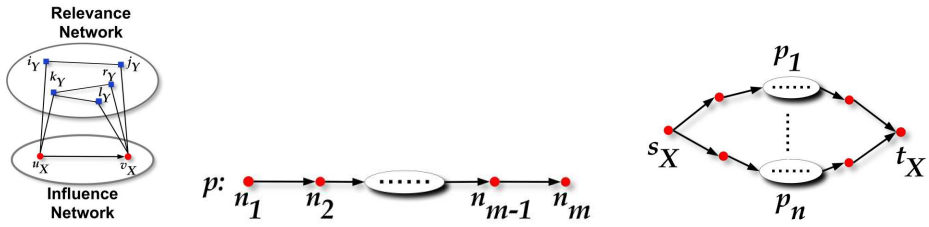
$$h_1(u_X, v_X) = 1 - \prod_{\substack{i_Y j_Y \in E_Y \\ u_X i_Y, v_X j_Y \in E_{XY}}} (1 - g(i_Y, j_Y)) \tag{3.1}$$

- When two nodes n_1 and n_m are not direct neighbors but they are connected via a sequential path p (Figure 11(b)), then we can define

$$h_p = \prod_{i=1}^{m-1} h_1(n_i, n_{i+1}). \tag{3.2}$$

- At last, when we try to calculate $h(s_X, t_X)$, and if there exists n paths between s_X and s_Y, we also assume that all these paths are independent. Thus,

$$h(s_X, t_X) = 1 - \prod_{i=1}^{n} (1 - h_{p_i}) \tag{3.3}$$



(a) neighbor (b) sequential (c) multiple

Figure 11. Activation Paths

The major difference between the above formulation and traditional activation probability in IC model is the activation probability between neighbors. We do not use uniform distribution to draw a

number and fix it for every edge in the influence network. Other than that, we calibrate the activation probability for every edge based on their nodes' connections to the relevance network and how those connections interact in the relevance network. Most recently, a study from (29) has explored the way of tailoring edge-dependent activation probability for influence maximization. However, it has no consideration of relevance information from another network, and it has no intention to use the reinforcement of influence and relevance information.

Now we will discuss how we define relevance scores in the Relevance network based on influence scores in the Influence network.

3.2.3 Relevance Measure on Relevance Network

We want to first clarify that relevance in this paper is an **asymmetric** concept that is different from the concept of node similarity. The similarity score of two nodes is usually symmetric and it has valid meanings in many applications. Relevance is a complementary concept to similarity because there also exists many other practical circumstances where $g(u_Y, v_Y) \neq g(v_Y, u_Y)$. For example, in a paper citation network we can only talk about papers' relevance when we are considering the relation between a paper and its citations. Relevance can only be counted from citations to the paper but not the other way around, since we consider using this relevance as a kind of influence. In a social network like Facebook, entities' relevance are also often asymmetric. For example, the way music being relevant to movie is often different from the way a movie being relevant to music. Therefore, we model the relevance in an asymmetric way.

Given two nodes $u_Y \in V_Y$ and $v_Y \in V_Y$ in the Relevance network $G_Y(V_Y, E_Y, F_Y)$, let $g(u_Y, v_Y)$ be the relevance of u_Y and v_Y. When defining relevance score of two nodes, we explore a similar

manner of the state-of-the-art link based on similarity research (39)(94) by considering the interactions of common nodes connected to them. SimRank (39) essentially considers common neighbors of two nodes as a starting point of their similarity measure. It goes through an iterative process to update such similarities based on the updated similarity values of other nodes in the network. PRank goes one step further by considering in-degree similarity and out-degree similarity in a directed network. PRank also utilizes an iterative process because two nodes' similarity depends on other nodes' similarities as well.

The major differences between our relevance computation and PRank's similarity computation lie in two folds. First, we let influence values from another type of nodes contribute in relevance computation, in addition to link-based analysis of PRank. Second, when considering the relevance of two nodes, for each node in one node's neighbor set, we take the maximum relevant node of it in the other node's neighbor set, rather than computing pair-wise similarities as PRank does. Figure 12 gives an example that illustrates the whole relevance computation. A_1 and A_2 are sets of nodes of in-links of i_Y and j_Y respectively, while B_1 and B_2 are sets of nodes of their out-links. In our model, $g(i_Y, j_Y)$ is related to $g(a_1, a_2), \forall a_1 \in A_1, a_2 \in A_2$ and $g(b_1, b_2), \forall b_1 \in B_1, b_2 \in B_2$. Furthermore, as we see i_Y and j_Y are connected to nodes in Influence network (represented in circle), the influence among u_X, v_X, z_X and w_X also contribute to $g(i_Y, j_Y)$. We use a weighted sum of two different parts to merge information from the Relevance network and the Influence network to compute $g(i_Y, j_Y)$. The first part is another weighted sum of the relevance of i_Y, j_Y's in-links set and the relevance of their out-links set in the Relevance network. The second part is the influence between the connected nodes of i_Y and connected nodes of j_Y in the Influence network.

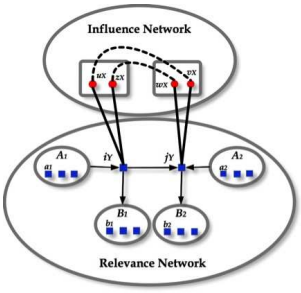


Figure 12. Relevance Example

Formally, the relevance between i_Y and j_Y is defined as follows.

$$g(i_Y, j_Y) = \sigma((\lambda \frac{1}{|I(i_Y)|} \sum_{\{k|(k,i_Y)\in I(i_Y)\}} max_{\{l|(l,j_Y)\in I(j_Y)\}} g(k,l)$$

$$(1-\lambda)\frac{1}{|O(i_Y)|} \sum_{\{k|(k,i_Y)\in O(i_Y)\}} max_{\{l|(l,j_Y)\in O(j_Y)\}} g(k,l))$$

$$+(1-\sigma) max_{\substack{z_X i_Y \in E_{XY} \\ w_X j_Y \in E_{XY}}} h(z_X, w_X) \tag{3.4}$$

Here $h(z_X, w_X)$ is the influence of z_X on w_X, and $I(i_Y)$ and $O(i_Y)$ are i_Y's in-degree and out-degree neighbors.

Up to now we have formulated the influence and relevance together on IR network. In the following subsection we will discuss how to compute them.

3.2.4 Iterative Procedure for Influence and Relevance Computation

Before diving into detail computations, we first briefly review our ultimate goal. Out of many practical demands we reduce a heterogeneous network to an IR network where we want to compute

influence maximization on Influence network and relevance analysis on the Relevance network. We have observed the benefit of passing information between these two tasks so that the results' qualities of both of them can be improved. We have also formulated new activation probabilities for influence maximization purpose and new relevance measures for relevance analysis. Both formulations originate from state-of-the-art works in these two fields.

From Eq. 1 and Eq. 2, we know that there is no close form solutions for each individual activation probability h and relevance score g, because they are nonlinear, non convex, and mutually dependent. Due to the fact that h and g are intractable, we design an iterative procedure to approximate their values.

Influence Expectation. On the Influence network, we follow the IC model to select initial seed nodes to maximize information spreading. Regardless of how we define activation probabilities, influence maximization is an NP-hard problem (44). We use the state-of-the-art greedy algorithms to do the seed selection (16). Every round, greedy algorithm selects a node which has the max margin gain as the next seed. In another word, the seed selected in every round is the one activates most other nodes. To determine whether a node could activate another, however, is a non-trivial task. There is no closed form to quantify the activation probability between every pair of nodes because multiple paths may exist between them and the paths may overlap. Therefore, activation simulation is the main method to compute node activations. We also use the expected value out of N iterations of simulations similar to (44)(16) (15), so we call this procedure influence expectation.

Assume that we have initial values of relevance scores of nodes from the Relevance network. We can calculate activation probabilities for node pairs on the Influence network. After this initialization, we take the expectation on activation results from many rounds of simulations to select the set of most

influential seeds. After the set is obtained in this manner, we head onto the analysis of relevance for nodes on the Relevance network.

Relevance Maximization. The relevance score from one node to another depends on the relevance scores of their neighbors. Computing the score is an iterative task because their neighbors' relevance scores may change during the iterations. Taking citation network as an example. Initially, relevance score of two papers a_Y and b_Y may be 0 if their co-citation list and bibliographic-coupling list are both empty. However, these lists being empty does not mean neither every paper referencing a_Y has nothing to do with the papers referencing b_Y nor papers cited by a_Y have nothing in common with b_Y's citations, since these papers have their own co-citation and bibliographic-coupling lists that may have some overlaps. As their relevance scores change in the iteration, a_Y and b_Y's relevance score may change. Moreover, the relevance score is also impacted by influence relations in the Influence network in our formulation. As the influence scores change among the authors of a_Y and b_Y, the paper relevance changes. Due to the nature of our model, these changes are monotonically increasing to different maximum values. Therefore, we name this step relevance maximization. Every time it first waits until the influence scores from the Influence network fully update and refreshes all the relevance scores. The influence expectation and relevance maximization procedure keeps running until both types of scores converge.

3.2.5 Details implementation and optimization for IR scheme

Here we discuss the details on the computation of the vectors of relevance scores g and influence scores h respectively. The whole idea is to first initialize g_0 using PRank 1^{st} round's score without influence information, feed g_0 to influence expectation to get h_0 and use h_0 for relevance maximization

to get g_1 and iterate this process until g and h converge. As we know, both influence maximization and relevance analysis are computation intensive tasks. Therefore, we further design some pruning methods to improve the efficiencies. For the sake of simplicity, nodes variables in this section no longer have subscripts indicating which graph they are in. This will cause no confusion since we only compute g function on Relevance network and h function on the Influence network.

3.2.5.1 Computation of g_0

As we stated before, $g(u, v)$ is the relevance of u to v in the Relevance network. Initially, we use the 1^{st} round's PRank of u and v to be $g_0(u, v)$. In order to facilitate the computation, we maintain two lists tracking the relationship between u and v, one for their in-links and the other for their out-links. With the help of these two lists, we can easily calculate the PRank values we need, and then we can get $g_0(u, v)$.

3.2.5.2 Computation of function h's value

Let $h(x, y)$ be the influence from x to y in the Influence network. After getting each round of function g value in the Relevance network, we use these values to compute h value of the current round under the Independent Cascade model. Firstly, we introduce $p(x \rightarrow y)$ ($\forall x, y \in V_X, \exists\ u, v\ xu \in E_{XY}, yv \in E_{XY}\ and\ uv \in E_Y$) representing the *direct* activation probability from x to y. The initial values of all $p(x \rightarrow y)$ are 0. Second, we consider each edge uv in the Relevance network as an independent activation path for the nodes x and y (if edges xu, yv exist) in the Influence network. Therefore when we go through the list of g and take each $g(u, v)$ into consideration, we update $p(x \rightarrow y)$ by $1 - (1 - p(x \rightarrow y))(1 - g(u, v))$ (for all $xu, vy \in E_{XY}$). After enumerating all $g(u, v)$, we get the

final direct activation probabilities between authors, and we can start the R rounds of simulation to get the approximate activation probability including *indirect* ones.

For the simulation, we define $t(x \rightarrow y)$ as the total times of x successfully activating y in all R simulations. Initially, $t(x \rightarrow y)$'s values are 0. In each iteration, we generate a random number between 0 to 1 to each path $x \sim y$ in author's network. If it's less than or equal to $p(x \rightarrow y)$, we consider this edge is activated. In each round of simulation, if x can reach y through activated edges, we add 1 to $t(x, y)$. After R simulations, we finally get the approximate value of $h(x, y) = t(x, y)/R$. The reason why we use this approximate way to calculate $h(x, y)$ instead of applying Eq. 3 is that is that one cannot generate a uniform equation to calculate the actual activation probability between two nodes for the overlap of nodes in different paths (44)(16).

3.2.5.3 Update of function g

After we get the value of function h, we can update the value of function g by using Eq. 4. The new value comes from two different parts: current g and h value. We update $g(u, v)$, if and only if they are direct neighbors in the Relevance network. One should notice that the structural difference between our IR algorithm and PRank is that we don't need to compute the values for all possible pairs of nodes, since we are only interested in those pairs which have a direct relevance relationship. After we successfully update the g values, we can apply the process described in the last paragraph to compute the new h values. We iteratively use the computation of h value and update g value until both of them converge.

3.2.5.4 Pruning and Dampening

As we know, both PRank and IC model are computational intensive. Therefore, in the design of our IR framework, we consider various of pruning methods. In the design of our relevance computation, we

have already mentioned that we only consider relevance among direct neighbors, although the iterative computation is through the whole network. That will reduce the relevance computation significantly.

In the influence maximization part, we focus on how to reduce activation simulation because it is the most time consuming component. Since each article usually contains more than two authors, once the g score between two papers turns from zero to a non-zero value, it can change significant numbers of h scores between authors from zero values to non-zero ones. However, the change of g score may be very small, this is because of when two papers has no overlaps in their citation lists (both forward and backward), the contribution to this change only from their authors' influence is very limited. As a result, after each iteration, many new edges may be added to the influence network when computing new h scores, and many of them are extremely small values. In practice, we found out the increase of these edges makes the algorithm very time consuming, and more than half of the weights of these edges are very close to zero. We consider these edges are not important, since it can change little to the h scores between authors. Therefore, we decided to apply pruning to make the computation more efficiently. Firstly, before we start R time's simulation, we remove those edges whose activation probability are below the average of all in the network, In addition, after we get the h value, we set those $h(x, y)$ who are below the $avg(x)$ to be zero, while $avg(x) = \frac{\sum_j h(x,j)}{\#non-zero\ h(x,j)}$. On the other hand, we also apply dampening process in the computation of $p(x \to y)$: when we take $g(u, v)$ into consideration, if $g_0(u, v) \neq 0$, update $p(x \to i) = 1 - (1 - p(x \to i)(1 - g(u, v)))$ ($i \in I(v)$, $x \in I(u)$), otherwise, we update $p(x \to i) = 1 - (1 - p(x \to i)(1 - g(u, v)e^{-g(u,v)}))$.

The whole algorithm is summarized below.

3.3 Evaluation

3.3.1 Dataset description

We use paper citation network (78) as an instance of IR network. The dataset has majority of papers in ACM digital library until 2008 with their citation relations and author information. We treat the paper citation network as Relevance network. If paper A cites paper B, we know that the authors of A have influence on authors of B. Therefore, we also construct an Influence network of author relations. In total, the Relevance network has $217,335$ nodes and $632,751$ edges, while the Influence network has $250,566$ nodes and $1,486,909$ edges. The number of edges between Relevance network and Influence network is $518,358$.

In the following, we will demonstrate the performance of combining relevance analysis and influence maximization together by comparing it with the state of the art social influence maximization algorithm (44)(16) on influence part and relevance computation algorithm (94) on the relevance part, respectively. On the activation simulation we set the simulation times $R = 1000$ for our method and both baselines (described below). We choose λ to be 0.5 and σ to be 0.8 for the IR model. In practice, our method converges on g and h values after 10 iterations. The experiment system is implemented in Java with JDK1.6, Eclipse and conducted on machines with Quad Core CPU with 2.2 GHz and 4GB RAM.

3.3.2 Comparing Seed Quality in Social Influence Maximization with State-Of-The-Art IC Model

3.3.2.1 Baseline Description

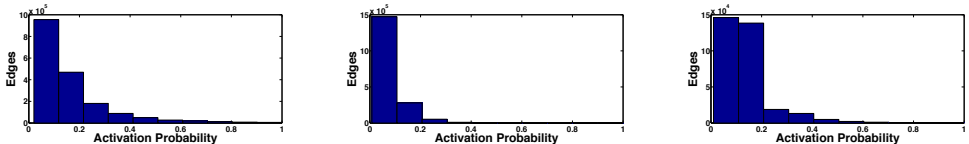
First of all, let us briefly review the goal of influence maximization research again. Given a population of users and their sharing behaviors for an information entity, e.g., an event, a product, the goal of social influence maximization is to find k users as seeds within the population, so that, suppose we pass the information entity to these k seeds, we end up having the most people activated among all the users (20)(72). Majority of influence maximization research did not focus on the network construction or the activation probability definition. Usually most of them are more interested in the algorithm of how to choose these seeds (44)(16)(53)(15). In order to compare the results of different algorithms, researchers normally analyze them under a same diffusion model (influence model). However, they rarely mention whether the final picked seeds are indeed the "influential" users in the real social network. This is because of that the previous works emphasize more on the design of the algorithms, and it's almost impossible to use real world data to verify the influence of the combination of a group of users given a certain incident. The independent cascade (IC) model is one of the origins of such research. It presumes that each user can try to activate its neighbors through several independent edges (connections) (44). Moreover, IC model and its variation Weighted Cascade (WC) model both assume that the numbers of the edges between two users depend on the times of previous successful passages of the information between them. As a result, they differentiate users' influential power by assigning different numbers of edges between different pairs of users, i.e. if a user is more influential to another user, it will have more edges connecting to that user. However, without further information, they can only randomly assign

the activation probability to each single edge. Instead of assigning activation probabilities like this, we claim that we are able to calibrate the activation probabilities in Influence Network using information from Relevance Network. Such calibration leads to better seeds quality and larger coverage. That is the reason we maintain the same diffusion model as IC to demonstrate the gain of more reasonable probability assignment and network construction. Because of the inefficiency of the original greedy algorithm (44), we implemented the approximate greedy algorithms in (16) to select seeds and follow the independent cascade model to stimulate activation process. We applied this same algorithm on our calibrated network and two other baseline networks to demonstrate the seeds from our network have higher qualities based on the widely accepted standards.

We compared with two baseline methods. The first one is the classic way to assign activation probability in original IC model, which is uniformly drawing a probability depending on the number of edges between two nodes. Despite the simplicity, it doesn't differentiate any edge. We will show that, with the help of the Relevance network information, we may assign more reasonable probability to each of these edges in order to pick up more reasonable "influential seeds". Therefore, in our method, activation probabilities are not uniformly distributed. To make a fair comparison, we control the median of the uniform distribution to be the same as the median of our distribution.

We also observe that the activation path structures change a lot from the classic baseline to our method due to our non-uniformed activation probability distribution. By comparing with the classic baseline, we will show that this new activation path structure is more reasonable. Furthermore, we want to show that each assignment of the activation probability on each path is also reasonable. Therefore, we design the second baseline as follows. We first obtain a distribution of our activation probabilities.

Secondly, by following that distribution, we generate a random number as the activation probability for every edge. Therefore, this generated network has the same path structure and activation probability distribution, but different assignment of each edge. We use it as our second baseline. Since this baseline has the same activation path structures as IR's, but more random probabilities, its performance should be in between of IR and original IC and closer to IR because it is IR's variation. The activation probability distributions of IR and two baselines are shown in Figure 13.



(a) IR (b) Generated (c) Original

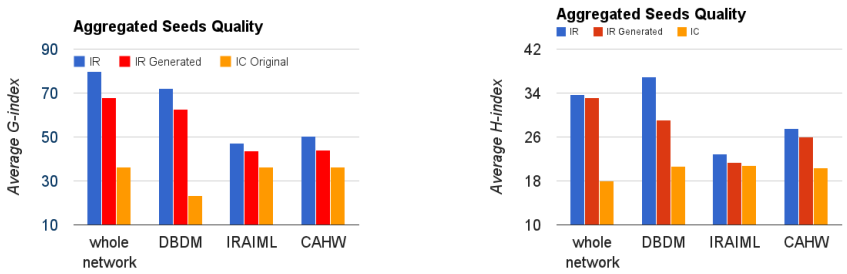
Figure 13. Activation Probability Distributions

3.3.2.2 Seeds Quality Comparison

In majority of social influence maximization research, seeds selection is the origins of their motivation. However, few works actually discussed what kinds of people those seeds are. They simply stated that they found k of them from whom the influence spreading would be maximized, and they paid more attention to discuss the performance of total number of activated nodes from those seeds. We think that it is important to give a throughout analysis on seed quality, since the quality is the key factor in real world applications of influence maximization. Therefore, we will first show aggregated comparison of

the seeds and then go to details of case studies to list the name of authors we identified as seeds and compare our seed list's quality with baselines. Through such comparison, we could demonstrate the IR computation generates more reasonable seeds.

Figure 14(a) shows the average G-index (21) of our seed list and the two baselines. Out of 40 seeds, we achieve an average G-index of 79.71 in whole Influence Network. That means the "seed" authors we identified have an average G-index of nearly 80, which is much higher than average G-index 36.37 by original IC baseline. If we introduce more randomness by generating activation probabilities according to the IR activation probability distribution, the result is in between of IR and original IC which is 67.89 by baseline 2. In addition to the whole area Influence Network, we extracted author relations within three sub-areas separately, including Database and Data Mining (DBDM) area , Information Retrieval, AI, and Machine Learning (IRAIML) area, and Computer Architecture and Hardware (CAHW) area. Similar performance also appear in these sub-area networks.



(a) G-Index (b) H-Index

Figure 14. Seeds Quality Comparison

Seed Quality Case Study Table VIII lists the top seeds selected on the whole Influence network equipped by our activation probabilities, and those probabilities provided by the two baselines. As we can see, when the activation probabilities are obtained through the calibration by information from Relevance network, we pick highly influential scientists from Computer Vision, AI, Machine Learning, Algorithms, and Information Security. The choices of top seeds from original IC (third column) contain more randomness, although their seeds are also reputable researchers. The generated scheme (IR variation) produces results in between IR and original IC, and its seeds qualities are closer to IR than IC's.

IR	IR Generated	IC Original
Andrew Zisserman	Michel X. Goemans	Shinto Eguchi
John McCarthy	Andries van Dam	Jianer Chen
Geoffrey E. Hinton	John McCarthy	Tibor Bosse
Charles E. Leiserson	Bruce Schneier	Matthias Gries
Bruce Schneier	Danny Raz	Daniel Cremers

TABLE VIII

TOP SEEDS IN WHOLE NETWORK

Similar effects happen in sub-areas in the whole Influence network (Table IX, Table X, Table XI). We always notice that seeds selected with activation probabilities produced by IR method are more influential than the two baselines. Randomness in selection increases in two baselines respectively. This

phenomenon confirms that when studying social influence maximization problem, carefully computation of reasonable activation probabilities can significantly improve both coverage and initial seeds qualities. We have demonstrated the advantage of using information from Relevance network to help achieve better influence maximization on the Influence network from the above aspects.

IR	IR Generated	IC Original
Jon Kleinberg	David Harel	Xuanhui Wang
Philip S. Yu	Masaru Kitsuregawa	Qiaozhu Mei
Jennifer Widom	Eugene Shekita	Li Chen
Michael Stonebraker	Philip S. Yu	Daniel Kifer
John A. Stankovic	Rajeev Motwani	Jimeng Sun

TABLE IX

TOP SEEDS IN DBDM

3.3.3 Comparing Relevance Computation with PRank model

We have seen the gain of combining influence maximization with relevance network information. Now we switch our attention from Influence network to Relevance network to explore how this combination gives better relevance values for node pairs in Relevance network. At the end of our IR computation, we have relevance scores for each pair of nodes in Relevance network. Under the same network, we

IR	IR Generated	IC Original
Thomas Hofmann	Norbert Fuhr	Norbert Fuhr
David R. Karger	Thomas Hofmann	Tao Tao
W. Bruce Croft	Katia Obraczka	Thorsten Joachims
Stephen Robertson	John Lafferty	Chengxiang Zhai
Gerard Salton	Justin Zobel	Mark Sanderson

TABLE X

TOP SEEDS IN IRAIML

IR	IR Generated	IC Original
David A. Patterson	Douglas C. Schmidt	Sarita V. Adve
A. Chandrakasfan	A. Zelikovsky	Rob A. Rutenbar
A. S. Vincentelli	Sartaj Sahni	Scott Rixner
Randal E. Bryant	G. D. Micheli	Frank Vahid
Robert K. Brayton	Massimo Poncino	Ravi Rajwar

TABLE XI

TOP SEEDS IN CAHW

use PRank (94) to compute another set of relevance scores [1]. We compare the two sets of relevance scores through clustering results and case studies. This comparison method has been frequently used in relevant research (94).

---

[1] "Relevance" is called *similarity* in (94) because their scores are defined symmetrically. Nevertheless, this difference does not affect the comparison here.

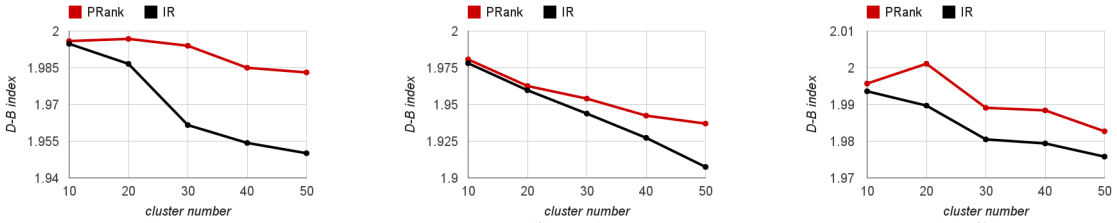
Cluster Compactness. We use $k\text{-}medoids$ to cluster nodes in Relevance network. Since we have relevance scores of nodes, we plug those scores as similarity measure into the clustering method. The goal is to see which set of scores produce higher quality clusters. We use the compactness of the result clusters as such quality measures. We use three sub-areas Relevance network, i.e., DBDM, IRAIML, CAHW, to perform the clustering procedure. We compute the compactness using Davies-Bouldin index (18). The total compactness of a clustering result is defined as

$$C = \frac{1}{n}\sum_{i=1}^{n} max_{i \neq j}\left(\frac{\sigma_i + \sigma_j}{d(c_i, c_j)}\right)$$

where c_i is the medoid of cluster i, $d(c_i, c_j)$ is the distance between two medoids c_i and c_j, and σ_i is the average distance of all nodes in cluster i to medoid c_i. σ_j has the same meaning in cluster j.

Their compactness results are shown in Figure 15. The relevance scores produced by IR scheme performs consistently better than PRank. As we intuitively discussed before, information from Influence network helps calibrate the relevance scores in Relevance network. As such an instance, two papers' relevancy is determined not only by their structural similarities, e.g., co-citation and bibliographic coupling, but also by their authors' relations. The intuition here is also straightforward: If author A could "activate" author B, B's papers may be more relevant to A's papers.

Relevance Paper Case Studies. To further explore the advantage of relevance scores generated by IR method, we pick several representative research papers in data mining field and list some of their citations ranked by two criteria, the relevance scores generated by IR method and PRank respectively. Our goal is to see which scoring method gives the most relevant citations the highest scores. Due



(a) DBDM (b) IRAIML (c) CAHW

Figure 15. Compactness Comparison in Relevance Networks

to the nature of our dataset, we are not able to select recent important papers but the classic ones. We choose 5 important papers related to frequent pattern mining, privacy-preserving data mining, and information retrieval that the readers are very likely to be familiar with. Therefore, people would know which citations are truly important to the 5 papers in advance, which leads to an easier judgment of the relevance score qualities.

Table XII summarizes the case studies. For the listed papers, most relevant papers ranked by IR scheme usually cover the real important references. Although top picks by PRank also contain important references, we can see there is more randomness in its selections. For example, on paper "An effective *hash*-based algorithm for mining association rules" the PRank missed an important reference of *hashing* mechanism but picked a general reference "Knowledge Discovery in Databases: An Attribute-Oriented Approach". Similar phenomena happen to other papers in the table. Although we could not go after every paper to see how IR scheme and PRank perform on picking most relevant references, we may observe that relevance scores generated by IR are more fine-tuned because of the input from Influence network.

paper	An effective hash-based algorithm for mining association rules, J.S. Park, et.al, SIGMOD'95
IR top relevant citations	**Fast Algorithms for Mining Association Rules in Large Databases, R. Agrawal, et.al, VLDB'94**
	Mining association rules between sets of items in large databases, R. Agrawal, et.al, SIGMOD'93
	File structures using hashing functions, E.G. Coffman, et.al, CACM'70
PRank top relevant citations	**Fast Algorithms for Mining Association Rules in Large Databases, R. Agrawal, et.al, VLDB'94**
	Mining association rules between sets of items in large databases, R. Agrawal, et.al, SIGMOD'93
	Knowledge Discovery in Databases: An Attribute-Oriented Approach, J. Han, et.al, VLDB'92

paper	Mining frequent patterns without candidate generation, J. Han, et.al, SIGMOD'00
IR top Relevant citations	**Fast Algorithms for Mining Association Rules in Large Databases, R. Agrawal, et.al, VLDB'94**
	An Efficient Algorithm for Mining Association Rules in Large Databases, A. Savasere, et.al, VLDB'95
	A tree projection algorithm for generation of frequent item sets, R.C. Agarwal, et.al, JPDC'00
PRank top relevant citations	**A tree projection algorithm for generation of frequent item sets, R.C. Agarwal, et.al, JPDC'00**
	Scalable Techniques for Mining Causal Structures, C. Silverstein, et.al, VLDB'98
	Efficient mining of emerging patterns: discovering trends and differences, G. Dong, et.al, KDD'99

paper	Privacy-preserving data mining, R. Agrawal, et.al, SIGMOD'00
IR top relevant citations	**A data distortion by probability distribution, C.K. Liew, et.al, TODS'85**
	Security-control methods for statistical databases: a comparative study, N. R. Adam, et.al, CSUR'89
	The statistical security of a statistical database, J.F. Traub, et.al, TODS'84
PRank top relevant citations	**A data distortion by probability distribution, C.K. Liew, et.al, TODS'85**
	SPRINT: A Scalable Parallel Classifier for Data Mining, J.C. Shafer, et.al, VLDB'96
	Security-control methods for statistical databases: a comparative study, N. R. Adam, et.al, CSUR'89

paper	Graph indexing: A frequent structure-based approach, X. Yan, et.al, SIGMOD'04
IR top relevant citations	**CloseGraph: mining closed frequent graph patterns, X. Yan, et.al, KDD'04**
	State of the art of graph-based data mining, T. Washio, et.al, KDD Explore'03
	gSpan: Graph-Based Substructure Pattern Mining, X. Yan, et.al, ICDM'02
PRank top relevant citations	An apriori-based algorithm for mining frequent sub-structures from graph data, A. Inokuchi, et. al, 2000
	CloseGraph: mining closed frequent graph patterns, X. Yan, et.al, KDD'04
	Algorithmics and applications of tree and graph searching, D. Shasha, et.al, PODS'02

3.4 Related Work

Influence maximization problem is an important research branch on social networks. The task is given a certain social network, how to choose some seed nodes (users) to spread certain information as wide as possible through mouth-to-mouth effect. It is first proposed in (20)(72) and officially defined and thoroughly analyzed by David Kempe et al (44), where independent cascade model and greedy algorithm are proposed to simulate and optimize the information spreading. Since then, more efforts have been devoted to either speed up the optimization process (16)(53) or improve the influential model (16)(15)(46)(80). Majority of them do not pay attention to how to get activation probabilities, which is one side of the problem we are resolving in this paper. Amit et.al (29) did the first work to attack this open problem. Their model is to learn the activation probability from action logs in social network. Our work is different from theirs in two ways. First, we do not use any guidance as action logs. Second and more importantly, their work does not consider heterogeneous network and does not use relevance measure to calibrate activation probability results. Another work (59) does consider social influence on heterogeneous network. However, its main focus is topic-level influence, i.e., how topics distribute over a heterogeneous network and how strong the influence among people on a specific topic. Our focus of improving influence and relevance analysis together has little overlap with theirs.

Measuring how nodes in networks relate to each other is another important research field in recent years. Nodes relations are often captured by relevance or similarity. In some papers, relevance and similarity are interchangeable concepts (77), or relevance is specially referred as the "similarity" of different types of nodes in heterogeneous networks (75). Similarity analysis on social network is usually based on nodes' common neighbors or link properties (45)(33)(39)(94)(58). In our relevance analysis,

we go down the path of quantify relevance using common neighbors. It is similar to (39)(94). However, we refer our analysis as relevance rather than similarity because we define it asymmetrically. Usually similarity between two nodes is a symmetric concept. Nevertheless, as we have discussed before, asymmetric concept is often needed. Moreover, we apply more information from social influence side to improve the relevance measure, which is not considered in previous similarity or relevance research.

We have also noticed that there is another work studied a total different relation of social influence and similarity together (17). They studied how people's influence and their similarities affect each other. In another word, they consider the same type of nodes' similarity and influence in a homogeneous network. We consider influence of one type of nodes, with the information of relevance of another type of nodes, and vise versa, in a heterogeneous network.

What we present in this paper starts from an unique observation that combining social influence and relevance analysis could benefit both of them. The main contribution we have that is to pursue this mutual beneficial relationship on a heterogeneous networks and design reasonable framework to exchange information between social influence and relevance analysis.

There is another line of research on bibliometrics is related to ours only because they are also studying the citation networks. Some of them aim at identifying important papers or authors in the citation networks using PageRank (68) or Authority-Hub(48) related techniques. Others explore features on node or edge similarities (31)(34)(60)(78). Moreover, some studies also explore topic level influences among research papers and their citation (59)(32). Both their major research topics and solution methods are far from the materials presented in this paper.

CHAPTER 4

MAGNET COMMUNITY IDENTIFICATION ON SOCIAL NETWORKS

4.1 Introduction

Community is an important building-block of social networks. It attracts people with common backgrounds, goals or interests, and it is the key element of social media's dominating success. For instance, college-level social community leads to the success of social-media giant Facebook; company-level social community initiates the popularity of professional network LinkedIn; business-level social community boosts the development of peer-to-peer business such as eBay. Moreover, in academic field, countless communities are formed to improve the academic society, such as the special interest group (SIG) for computer scientists, the Society for Industrial and Applied Mathematics (SIAM), etc.

Among all social communities, we notice that some of them attract people's interests more than the others, and we call them *magnet communities*. More specifically, *magnet communities* are such communities that draw significantly more attention than others even if they are all about the same topic. Examples of magnet communities include the magnet colleges with over 20,000 talented applicants each year (e.g., Stanford, MIT, etc), magnet research communities with over 1000 high quality paper submissions (e.g., KDD), and magnet companies professionals would love to work for. Communities draw people's attention from each other and form an interesting network. For example, Figure 16 depicts such a network where the employees migrate among some prestige IT companies/communities [1].

[1]The information is from (6)

According to this statistics, every time there is one person joining Google from Facebook, there are 15.5 people joining Facebook from Google. Though we are not discussing which company is "*better*" from the figure, we notice that Facebook is one of the *magnet* communities that people prefer to join.

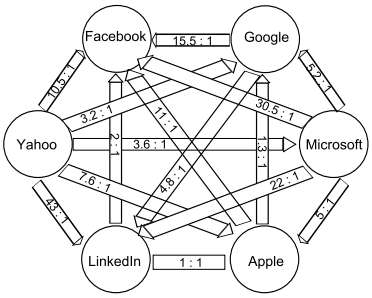


Figure 16. Community Interactions

Identifying magnet communities in a specific domain is an important task. First, the magnet communities help people understand the trends of their domains, e.g., which topics are hot, what kinds of new interests people are developing, and which types of new communities are forming. Second, they help people make decisions when joining communities. For example, irrelevant to communities' sizes or histories, people may want to join magnet ones that represent energetic present and promising future. These two functions of magnet community identification can further serve as building blocks of many other applications, e.g., community recommendation, etc. Despite its importance, magnet community identification problem has not been studied before, to the best of our knowledge. Researchers have focused on community detection in various scenarios. Given a network, their goal is to detect sub-

networks of similar nodes as communities. Our goal is one step further to identify magnet communities in a given domain through quantitative analysis. More specifically, given communities in a domain, we want to rank them based on their attractiveness to people among the communities of that domain. In the end, the top ranked communities are the ones people tend to adhere to.

To solve this problem, a quick path to go down is the classic random walk scheme. PageRank or its variations seems to be the simple and right choice. However, we demonstrate their incompetence on this problem using Table XIII, where we list 3 ranking results based on different mechanisms. The first column lists the companies with highest PageRank scores among 6853 IT companies in the employee migration network that we have. Interestingly enough, they are also the 5 largest companies in terms of employee numbers in our dataset. Therefore, standard PageRank skews heavily towards the size of a community. What about we simply normalize the migration flow by community size? The second column of Table XIII lists the results after normalization. The top ranked companies become tiny start-ups with about 100 employees. The third column are the companies recognized as "ideal employers" according to a survey result from Universumglobal [1]. It aligns better with people's common sense about IT industry. As we have seen, the classic random walk schemes cannot accurately measure the attractiveness of communities. Although the survey result can create better performance, it takes a lot of efforts and manual works, which becomes infeasible for large-scale identification tasks.

Therefore, the magnet community identification task is more challenging than it appears to be. First, there is no single criterion that we could rely on to determine the attractiveness of a community. For ex-

---

[1] www.universumglobal.com/Top50

Rank	PageRank	Normalized PR	Survey Result
1	Hewlett Packard	Zuora	Google
2	IBM	Silver Peak Systems	Microsoft
3	Oracle	Kony Solutions	Apple
4	Microsoft	Palo Alto Networks	Facebook
5	Cisco Systems	Quickoffice	IBM

TABLE XIII

TOP RANKED IT COMPANIES

ample, we cannot blindly say that a larger community is magnetic while a smaller one is not. Therefore, the first challenge is how to extract features from these heterogeneous sources of impacting factors of a community's attractiveness. The second challenge is how to naturally combine all heterogeneous information into a unified ranking model that also depicts attractive properties. For example, some features are only related to each individual community, e.g., size or specific topic, which have little to do with other communities; while others depend on other communities, e.g., people's preferences. The third challenge is the noise handling. People may have different tastes about communities. It is possible that a few people may prefer the less attractive communities. However, we are focusing on the major trend.

We attack the problem by first identify some common properties of magnet communities and design appropriate solution accordingly. First, we notice that a community's attractiveness associates with its *attention flow*. A community is attractive when it draws more people's attentions than losing attentions to other communities. In other words, the in-flow of people's attentions could be larger than out-flow for magnet communities. Second, a community's attractiveness associates with *attention quality*. It goes higher if it could draw people's attention from other magnet communities. Therefore, magnet commu-

nities could not only attract people, but also attract them from other attractive communities. Third, a community's attractiveness also associates with the *persistence* of people's attention. Being able to have the first two properties for only a short period does not make a real magnet community. When a community can persistently sustain the two properties, it becomes a magnet one. With these observations, we formulate the magnet community identification problem as a constrained graph ranking problem with heterogeneous information from nodes and edges, where constraints represent the relations between these features and magnet properties we just discussed. The whole model becomes an optimization framework of a synthesis measure of attractiveness of communities, which we will go into details later.

We summarize our contributions in three folds:

- We propose a new direction on social network analysis, namely magnet community identification. We believe that the work towards this direction is of huge demands of social network applications, since it can improve user experiences, increase their engagements to social network platforms, and enrich application features on these platforms.
- We propose one definition of magnet communities by identifying their properties. We develop a framework for large community graphs, which models both community features and the attractiveness properties together to identify magnet communities that reflect people's preferences among them.
- We demonstrate the effectiveness of our framework on a particular domain of magnet community identification, namely company's employee magnet community identification. As a tip of the iceberg of general magnet community identification, this domain specific example could serve as an

interesting topic by itself, which further proves the wide range of applications magnet community identification may have.

This chapter is organized as follows. In the next section, we formally define the magnet community identification problem and propose our framework in details. In section 3, we show our experiment results on real world social network data. In section 4, we explain the relationship between our work and previous related ones. Section 5 is the conclusion.

4.2 Magnet Community Identification Framework

Starting from a bird's-eye view of the magnet community identification framework, we will go deeply through the formulation of our model in this section.

We first represent a social network as a graph $G(V, E, f_V, f_E)$, where V and E are vectors of nodes and edges, and f_V and f_E are features of nodes and edges respectively. To deal with the large scale nature of social networks, we define a summarization function $\mathscr{L} : G \rightarrow \mathcal{G}_\mathcal{C}$ that maps an original graph to its condensed graph $\mathcal{G}_\mathcal{C}(\mathcal{V}, \mathcal{E}, \mathcal{F}_\mathcal{V}, \mathcal{F}_\mathcal{E})$, where $\mathcal{V} = \{\mathcal{V}_1, \mathcal{V}_2, ..., \mathcal{V}_k\}$ $(\forall i, \mathcal{V}_i \subset V)$ and $\mathcal{E} = \{\mathcal{E}_1, \mathcal{E}_2, ..., \mathcal{E}_n\}$ $(\forall i, \mathcal{E}_i \subset E)$ are subsets of nodes and edges of original graph. Moreover, $\mathcal{F}_\mathcal{V}$ and $\mathcal{F}_\mathcal{E}$ are aggregated nodes' and edges' features in the condensed graph. For the sake of modeling simplicity, we further require $\forall i, j, \mathcal{V}_i \cap \mathcal{V}_j = \emptyset$. Therefore, every element in $\mathcal{V}$ is a community, and every element in $\mathcal{E}$ is a hyper-connection between two communities (a set of edges across the two communities in original graph). Such connections could be defined in various ways, e.g., their domain relation, or people's preference on each other. Figure 17 illustrates such a compression process. Community detection algorithms could be applied in this process. However, it is not the emphasis of this chapter. We directly focus on magnet community identification on $\mathcal{G}_\mathcal{C}$.

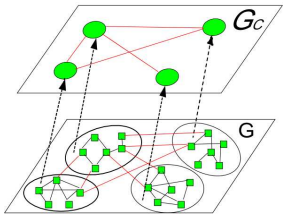


Figure 17. Graph Compression

4.2.1 Basic definitions

Generally speaking, magnet communities are the ones that people prefer to pay attention to among all social communities. In order to quantitatively identify magnet communities, we first give some more precise definitions related to them, from which we formally depict magnet communities later on in our model.

Definition 4.2.1 *(Attention to a community) People's attention to a community is a generic representation of their selectively concentration on the community while ignoring others.*

In other words, we assume one person only pays attention to one community at a given time. The person's attention may change from one community to another from time to time, but one at a time. Paying attention has generic meaning, e.g., working for a particular company, or submitting paper to a special interest group. When we mention that a community is observed to attract people's attention, we assume that the people being attracted only pay attention to that community but not others at the time of observation.

Definition 4.2.2 *(Attention migration flow among communities) Attention migration flow is the overall changes of people's attentions among communities, at the time of observation.*

When a person loses her interest in one community and starts to pay attention to another, there is an attention migration. For one community, its *in-flow* represents the total attentions drawn from other communities, and *out-flow* represents the total attentions lost to other communities.

Definition 4.2.3 *(Magnet community) A community is a magnet community, i.e., with high attractiveness level, if it has the following properties.*

- *Attention flow: Its in-flow is larger than the out-flow.*
- *Attention quality: The in-flow comes from other communities with high attractiveness levels.*
- *Attention persistency: Its first two properties should be persistent.*

The first property ensures that the size of a magnet community should be growing, because it could attract more and more people. The second property tells the fact that a community's attractiveness should depend on other communities as well. It is more attractive if it could draw people's attention over other attractive ones. The third property, persistence, guarantees that we identify true magnet communities rather than some communities only with a momentarily attention burst. We also notice that the community size does not directly contribute to the attractiveness value. However, magnet communities usually have a certain size that is neither too small nor too large. On the one hand, the persistence property will ensure an accumulative effect of community growth to make itself become larger. On the other hand, when the community has grown into a large one, it is hard to keep in-flow larger than out-flow. That is the reason that very large communities tend to become less attractive due to the loss of people's

attention. For example, one particular friendship community of a person may be more attractive when it contains certain number of friends sharing information, but when it grows too large and interesting information get diluted, it becomes less attractive.

4.2.2 Attractiveness computation framework

We use a vector $\mathcal{M} = (m_1, m_2, ..., m_k)$ to denote the set of estimated attractiveness levels for all communities. As we mentioned, attractiveness level of a community m_i is determined not only by heterogeneous features $\mathcal{F}_\mathcal{V}$ and $\mathcal{F}_\mathcal{E}$ from nodes and edges in the condensed community graph, but also by the attractiveness values of other communities in the same domain. Therefore, we have an abstract representation of $\mathcal{M}$ as

$$\mathcal{M} = f(\mathcal{F}_\mathcal{V}, \mathcal{F}_\mathcal{E}, \mathcal{M}) \tag{4.1}$$

Let $\mathcal{M}^*$ be the real attractiveness values. Our goal is to make the gap between the real values and the estimated ones as small as possible. Therefore, our objective function at high level is to minimize the Frobenius norm of the two, with the constraint φ to ensure that the final attractiveness levels for communities are consistent with their magnet properties.

$$min||\mathcal{M}^* - \mathcal{M}||_F^2 \tag{4.2}$$

$$s.t., \varphi(\mathcal{M}, \mathcal{F}_\mathcal{V}, \mathcal{F}_\mathcal{E}) \geq 0 \tag{4.3}$$

In the following subsections, we will concentrate on more details about feature extractions and concrete formula of our objective function and its constraint.

4.2.3 Attractiveness features

4.2.3.1 Standalone features

As the name indicates, a community's standalone features are those not depending on other communities. For example, the topic of a community is a standalone feature. For example, if a group on LinkedIn is about text mining, it is very unlikely that it changes that topic due to other groups' activities.

Suppose we have h standalone features $\Gamma = \{\gamma_1, ..., \gamma_h\}$, the score of standalone features of community i is a function of Γ_i. $f_i = g(\Gamma_i)$, where Γ_i is the values of the standalone features of community i. Therefore, $\mathcal{F}_\mathcal{V} = (f_1, f_2, ..., f_k)$ becomes the standalone feature vector for all the nodes in $\mathcal{G}_\mathcal{C}$.

4.2.3.2 Attention migrating matrix as dependency features

We call a community magnet because it can draw people's attention from other communities. Thus, a straightforward way to derive dependency features is to start from people's attention migration flow. People in one community may stay active in that community, or become inactive because some other communities draw their attention away, so that they become active in somewhere else. Therefore, communities draw people's attention among each other. This unique relation is modeled as an attention migrating matrix $D = (d_{ij})_{k*k}$, where d_{ij} is the actual number of people who *depart* from community i and *join* j. Departing from i means the person becomes inactive in community i. Respectively, joining j means the person becomes active in j. Note that what we assume here is that every person could only be active in one community at one time. Let vector $A = (a_i)_{k*1} = D \cdot e$ be the attention vector, where e is a k-by-1 unit vector. Thus, a_i is the total number of people who depart from community i. Let $\mathcal{A}$ be the element-wise inverted vector of A, where $\mathcal{A} = (a_i^{-1})_{k*1}$. We have dependency features of communities as $\mathcal{F}_\mathcal{E} = \mathcal{A} \circ D^T$, which is the Hadamard product of $\mathcal{A}$ and D.

The dependency matrix, or edge feature, $\mathcal{F}_{\mathcal{E}}$ is a probabilistic transitional matrix [1]. Each column of $\mathcal{F}_{\mathcal{E}}$ is the distribution of people whose attentions are migrating to other communities.

4.2.4 Concrete formula of magnet community ranking framework

One way to model the attractiveness of a community, i.e., a node in $\mathcal{G}_{\mathcal{C}}$ is to use random walk with restart. A node's attractiveness value depends on the probability of it being visited from other nodes. In other words, it is the probability that people's attention would arrive from other communities. Upon combining heterogeneous features about node and edge on $\mathcal{G}_{\mathcal{C}}$, we have

$$\mathcal{M} = \alpha\mathcal{F}_{\mathcal{E}}\mathcal{M} + (1-\alpha)\mathcal{F}_{\mathcal{V}}, 0 \leq \alpha \leq 1 \tag{4.4}$$

where α is a weighting parameter. With that formula, we can rewrite the objective function as

$$min\ ||\mathcal{M}^* - \mathcal{M}||_F^2 \tag{4.5}$$

$$=\ min\ ||\alpha\mathcal{F}_{\mathcal{E}}\mathcal{M} + (1-\alpha)\mathcal{F}_{\mathcal{V}} - \mathcal{M}||_F^2 \tag{4.6}$$

$$=\ min\ ||(\alpha\mathcal{F}_{\mathcal{E}} - I)\mathcal{M} + (1-\alpha)\mathcal{F}_{\mathcal{V}}||_F^2 \tag{4.7}$$

Now let us focus on the constraint for the above objective function. When we say one community is more magnetic than the other, at least one of the following two conditions are very likely to happen. First, this community has better standalone features. Second, it draws people's attention out of other

---

[1] If two nodes are unreachable, we add a small probability for them to directly reach each other to make $\mathcal{F}_{\mathcal{E}}$ irreducible and aperiodic. (PageRank computation has a similar procedure (67))

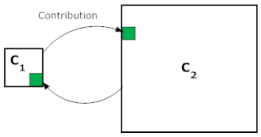


Figure 18. Contribution imbalance

similar communities. On the contrast, it is unlikely for a community to be more magnetic than others if it is inferior on both conditions. Formally, when i is more magnetic than j, i.e., $m_i - m_j > 0$, we want at least one of the following conditions hold.

- $f_i > f_j$
- $\frac{d_{ji}}{S_i} > \frac{d_{ij}}{S_j}$

The first condition is straightforward. It simply states that i's standalone feature should be higher than j's, if i has high attractiveness level than j. The second one needs more explanation. When people depart from community i and join j, we say that i has contribution to j. If community i is more attractive than j, j's contribution to i should be larger than i's contribution to j. The significance of this contribution depends on two factors: the number of migrated people from j and the size of i. Larger communities are harder to be contributed significantly than smaller ones with the same size of in-flow (see Figure 3). Therefore, we call the second condition *contribution imbalance*.

Although it is possible for people to move from i to j if only one of the above conditions is true, it is very unlikely for them to prefer i over j if none of the two condition is true. Thus, we make our constraint as follows, where μ is a weighting parameter and ζ is a lower bound.

$$\sum_{(i,j)} (m_i - m_j) * (\mu(\frac{d_{ji}}{S_i} - \frac{d_{ij}}{S_j}) + (1-\mu)(f_i - f_j)) \geq \zeta \tag{4.8}$$

Therefore, the three properties of magnet communities are captured into Eq. 7 and Eq. 8 in a subtle way. Eq. 7 states that a community would have better chance to be a magnet one if it attracts attentions from other high magnet communities, which implies the second property. Eq. 8 constraints the magnet computation results must consistent with the first and third properties in Definition 2.3, which are reflected by the term $\frac{d_{ji}}{S_i} - \frac{d_{ij}}{S_j}$.

We rewrite the constraint after combining like terms as

$$\sum_{i=1}^{n} \left(\sum_{u \in n_{in}^i} \varphi_{iu} - \sum_{v \in n_{out}^i} \varphi_{vi} \right) m_i \geq \zeta \tag{4.9}$$

where n_{in}^i is node i's in-degree neighbors in $\mathcal{G}_\mathcal{C}$ and

$$\varphi_{iu} = \mu(\frac{d_{ui}}{S_i} - \frac{d_{iu}}{S_u}) + (1-\mu)(f_i - f_u)$$

n_{out}^i is node i's out-degree neighbors in $\mathcal{G}_\mathcal{C}$, and φ_{vi} has similar meaning with φ_{ui}.

We organize Eq. 9 in a more abstract form as

$$\Phi\mathcal{M} \geq \zeta \tag{4.10}$$

Here, $\mathcal{M}$ is the vector of $\{m_i\}_{1*n}$ and Φ is its coefficient vector.

Now we discuss how to solve the optimization framework.

Theorem 1 *Our optimization framework is equivalent to the following canonical quadratic programming form:*

$$min\left(\mathcal{M}^T Q\mathcal{M} - 2u^T\mathcal{M}\right) \tag{4.11}$$

$$s.t., H\mathcal{M} \leq \xi \tag{4.12}$$

Proof 1 *The objective function of Eq. 7 can be rewritten as*

$$||(\alpha\mathcal{F}_\mathcal{E} - I)\mathcal{M} + (1-\alpha)\mathcal{F}_\mathcal{V}||_F^2$$

$$= tr(\mathcal{M}^T(\alpha{\mathcal{F}_\mathcal{E}}^T - I)(\alpha\mathcal{F}_\mathcal{E} - I)\mathcal{M} + (1-\alpha)\mathcal{M}^T(\alpha{\mathcal{F}_\mathcal{E}}^T - I)\mathcal{F}_\mathcal{V}$$

$$+(1-\alpha){\mathcal{F}_\mathcal{V}}^T(\alpha\mathcal{F}_\mathcal{E} - I)\mathcal{M} + (1-\alpha)^2{\mathcal{F}_\mathcal{V}}^T\mathcal{F}_\mathcal{V})$$

Notice that the first three terms here are single number variables, and the fourth term is a constant. Therefore, above objective equation is equivalent to the form below:

$$(\mathcal{M}^T(\alpha{\mathcal{F}_\mathcal{E}}^T - I)(\alpha\mathcal{F}_\mathcal{E} - I)\mathcal{M} + 2(1-\alpha){\mathcal{F}_\mathcal{V}}^T(\alpha\mathcal{F}_\mathcal{E} - I)\mathcal{M}$$

Let $Q = (\alpha {\mathcal{F}_\mathcal{E}}^T - I)(\alpha \mathcal{F}_\mathcal{E} - I)$, $u = (1-\alpha)(\alpha {\mathcal{F}_\mathcal{E}}^T - I)\mathcal{F}_\mathcal{V}$, $H = -\Psi$, *and* $\xi = -\zeta$. *We have our optimization framework reduced to the target form.*

Next, we explore the property of the solution of our optimization, i.e., whether it has feasible solution, or whether the global minimal exist, and the complexity of solving the optimization.

Lemma 1 *Q in the optimization framework is positive definite, if $\alpha \neq 0$ and $\alpha \neq \frac{1}{\lambda}$, where $\lambda \neq 0$ is the eigenvalue of $\mathcal{F}_\mathcal{E}$.*

Proof 2 *Since $\mathcal{F}_\mathcal{E}$ is a stochastic matrix, $|\mathcal{F}_\mathcal{E}| \neq 0$. We have $\mathcal{F}_\mathcal{E} X = \lambda X$. Therefore, $(\alpha \mathcal{F}_\mathcal{E} - I)X = \alpha \mathcal{F}_\mathcal{E} X - X = (\alpha\lambda - 1)X$. We have the eigenvalue of $(\alpha \mathcal{F}_\mathcal{E} - I)$ is $\alpha\lambda - 1$. Thus, $|\alpha \mathcal{F}_\mathcal{E} - I| = \alpha\lambda - 1 \neq 0$. Since $Q = (\alpha \mathcal{F}_\mathcal{E} - I)^T(\alpha \mathcal{F}_\mathcal{E} - I)$, Q is positive definite.*

Theorem 2 *Our objective function is strictly convex and it has a unique local minimum which is also a global minimum. Ellipsoid method can find global minimum of the objective function in polynomial time.*

Proof 3 *The convexity of objective function and the existence of global minimum are guaranteed by the positive definite Q, which also ensures there are polynomial time algorithms can find the minimum solution (12).*

With the solution techniques, attractiveness computation framework for communities can be formally illustrated below.

4.3 Evaluation

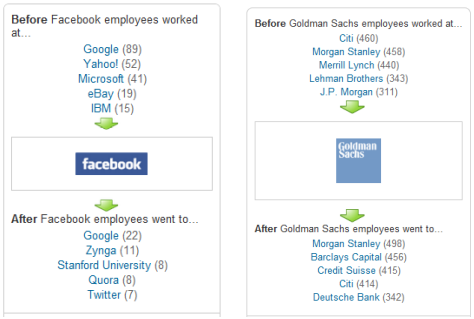
In this section, we apply the magnet community identification framework to one real world application, namely magnet company identification, on a social network of professional relations in two

industries. Magnet company identification is an interesting research topic by itself. As it serves as a good instance for our framework as well, we select this unique angle to show the effectiveness of our model. All experiments are conducted on machines with Intel Xeon^{TM} Quad-Core CPUs of 2.27 GHz and 24 GB RAM. We use the Matlab optimization toolbox as the solver for our framework.

4.3.1 Data collection and features extraction

4.3.1.1 Data collection

We crawl the company community data from the world's largest online professional network platform www.linkedin.com. It provides information about many aspects of companies, one of which is the employee flow (Figure 19). For most companies with certain size or reputation, LinkedIn lists up to



(a) Facebook employee flow (b) Goldman Sachs employee flow

Figure 19. Employee Migration Flows on Linkedin

top 5 companies of their employee sources and destinations. With this publicly available information,

we can construct the raw departure matrix. LinkedIn also displays an estimated size of every company. Therefore, we can piece the information together to have our contribution matrix, which serves as the dependency features. Although LinkedIn only contains partial information about these companies (not all the employees have LinkedIn accounts), it is a close estimation of real world situation and a good sample of the ground truth data.

For standalone features, we select a company's revenue per employee [1] (a commonly used factor to describe company efficiency (9)), industry, location, and age as its raw features. We believe the four factors contribute much to one's decision about joining a company.

Notice that we can directly consider the information we crawled the features on condensed graph $\mathcal{G}_\mathcal{C}$. In fact, in this application domain, the original graph G should be the whole professional network, including users features and their connections. Since that information is not publicly available, we could omit the graph compression process. However, we argue that the performance of our model may not be impacted significantly, because the publicly available information is already an aggregated representation of the hidden condensed graph we need. As a result, the different aspects of information about company communities we crawled can be plugged right in to the optimization framework. In total, we have 39527 companies' information in 142 industries.

---

[1] Since LinkedIn does not have this information, we crawl that from www.reuters.com/finance.

4.3.1.2 Feature extraction

Our raw features for community standalone features in this case are a company's revenue-per-employee, industry, location, and founded year. They are categorical valued except for the first feature. Now we discuss how we translate them into numerical forms.

For each industry, we count how many people flow into it and out of it, using company level departure and arrival data. With that data normalized by industry size, we have the percentage growth of each industry. This percentage indicates the attractiveness of the industries. Table XIV is such an example. It lists growth of all industries related to IT and Finance. We use this growth rate as the industrial feature value.

IT Industry	Growth	Financial Industry	Growth
hardware	3%	banking	1%
software	3%	insurance	0%
network	6%	finance	2%
internet	11%	real estate	0%
semiconductor	2%	investment banking	3%
telecomm.	-2%	investment mngt.	3%
electronics	0%	accounting	1%
information service	1%	venture capital	5%
online game	6%	commercial real estate	6%
information security	4%	capital markets	-1%

TABLE XIV

INDUSTRY GROWTH

For locations, we measure their popularizations. If a location has more companies nearby, it is more popular. For example, Silicon Valley and New York City are two popular places for IT and finance industries, respectively. A company should be more attractive if it is at a more popular location. We use the number of companies of a location as its feature. We further normalize it to the range (0,1).

Founded year feature is treated similar to location features. For example, people in high-tech industry would like to work for companies that are not only mature enough to be stable, but also not founded too long ago to be active. Thus, we count the number of companies founded for each year. We use normalized value to represent year feature.

After the categorical-numerical translation, we use a linear combination of the values from the four features as the standalone feature. The dependency features are from the transition data we mentioned in data collection section.

4.3.2 Ranking performance

4.3.2.1 Baseline Description

We choose PageRank with one simple but important variation as our baseline. We replace its originally binary adjacent matrix with $\mathcal{F}_{\mathcal{E}}$, and its uniform distributed restart vector with $\mathcal{F}_{\mathcal{V}}$. We still call it PageRank for the ease of presentation. However, this variation captures more information than original PageRank and slightly improves its results, so it serves better as a baseline.

It makes more sense to compare companies in the same industries. Therefore, we pick two representative industries, IT and financial, to conduct our experiments. Although there are divisions within each meta industry (e.g., IT has segments of software, hardware, etc), they are comparable because they share many common properties and attract similar employees. Since our objective is to identify *magnet*

companies, the desired result should be companies that have the most potential at current time, which will accelerate the new technology development, produce new ideas, and significantly change people's lives. They are not necessarily giant or well-established ones, but rather innovative and attractive to talents. We searched through media for such rankings as the ground truth. Unfortunately, despite many kinds of company rankings, there is no one directly based on the criteria we mentioned above. Take the arguably most recognized company ranking system *Fortune Magazine* as an example. The famous *Fortune 500* is for largest corporations, because it is based on size and revenue. Other simple rankings proposed by Fortune are based on straightforward criteria. For example, its "best companies to work for" is a series of rankings each of which is based on one of the conditions like *job growth, no layoffs, best for women, big pay*, etc. Out of our best effort, we found no single ranking that takes as many aspects of *attractiveness* level as we did. Nevertheless, we found two rankings that are related to our criteria. One such ranking is the 2011 *ideal employer ranking* proposed by *Universumglobal*while the other is the 2011 most admired company ranking by Fortune [1]. Both of them are based on survey results from professionals and university students. We will use them as our reference sets to compare the effectiveness of our results with PageRankresults. Before that, we would like to first demonstrate some interesting case studies. This provides intuitions and insights of the performance of our magnet community identification model, and why it identifies real magnet companies that attract talents.

---

[1]money.cnn.com/magazines/fortune/mostadmired/2011/index.html

Rank	PageRank	**MIM**	Ideal Employer	Admired Company
1	IBM	**Google**	Google	Apple
2	Hewlett Packard	**Amazon.com**	Microsoft	Google
3	Oracle	**Apple**	Apple	Amazon.com
4	Microsoft	**Microsoft**	Facebook	IBM
5	Cisco Systems	**Facebook**	IBM	Qualcomm
6	Google	**Salesforce.com**	Electronics Arts	Intel
7	Tata Consult. Services	**Cisco Systems**	Amazon	Texas Instruments
8	Cognizant Tech. Solu.	**Juniper Networks**	Cisco Systems	Cisco Systems
9	Dell	**Yahoo!**	Intel	Adobe Systems
10	EMC	**Linkedin**	Sony	Oracle

TABLE XV. Top 10 IT Companies

4.3.2.2 Case studies

First we want to be clear that, in this study, we are not judging which companies are better. We only neutrally comment on the trend of the companies attractiveness on talents.

IT magnet companies general trend. Table XV lists the top-10 ranked companies by PageRank and magnet identification model (MIM) results, along with the survey results from Universumglobal and Fortune about ideal employer and most admired companies, respectively. The magnet company list we identified contains both well-established companies, such as Google, Apple, and Microsoft, and energetic start-ups such as Facebook, Salesforce, and LinkedIn. As we know, the technology trend of IT industry is migrating from software to internet in the early 2000s, and from internet to mobile applications and social networks in recent years. The top magnet companies that we identified correctly reflect

this trend. For example, software company Microsoft has become less magnetic than internet company Google; Apple is attractive because of the popularity of mobile applications; and social network company Facebook is catching up with other prestigious companies on attractiveness level even if it was just founded a few years ago. In contrast, PageRank prefers big companies because they have far more employee flows than small companies. The PageRank's score depends heavily on a company's size, which is why IBM and Hewlett Packard outrank Microsoft, and Microsoft outranks Google. Compact yet competitive companies like Apple do not earn the chance to get into top 10. These clearly counter people's general understanding of these IT companies and the industry trend. Moreover, in its top 10 list, PageRank does not identify small but promising companies.

We also notice that our result has some important differences with the two survey results, even if our results are already closer to them than with PageRank's result. In the survey results, well-established companies trend to have higher ranks, maybe because people are more familiar with them. For example, companies such as Cisco, Intel, and Texas Instruments will be more recognized due to their legacies. However, well-established companies do not necessarily correspond to magnet companies. Sometimes small or medium start-ups represent the direction of brand new trends which may not even be realized by majority of the population taking the survey. Therefore, these kinds of surveys could not capture this phenomenon. That is why MIM works better than them. For example, we are able to identify promising companies such as Facebook, Salesforce.com and LinkedIn in top 10, and Twitter, Zynga in top 25, where they are hardly recognized by the survey results.

Micro-level case study It is interesting to note that our model, in its top 10 list, is able to identify all the companies in Figure 16 that is generated from a separate source (6). From that figure, we can see

Rank	PageRank	**MIM**	Ideal Employer	Admired Company
1	J.P. Morgan Chase	**J.P. Morgan Chase**	Goldman Sachs	US Bank
2	Citigroup	**Goldman Sachs**	J.P. Morgan Chase	Goldman Sachs
3	HSBC	**Morgan Stanley**	Boston Consult. Grp.	J.P. Morgan Chase
4	PWC	**Citigroup**	Deloitte	Merrill Lynch
5	Merrill Lynch	**Merrill Lynch**	Merrill Lynch	Northern Trust Corp.
6	Ernst & Young	**CB Richard Ellis**	Ernst & Young	Credit Suisse
7	Deutsche Bank	**Wells Fargo**	Morgan Stanley	CB Richard Eills
8	Credit Suisse	**PWC**	PWC	HSBC
9	Barclays Capital	**Jones Lang LaSalle**	American Express	Barclays
10	Goldman Sachs	**Blackrock**	Bain & Company	Jones Lang LaSalle

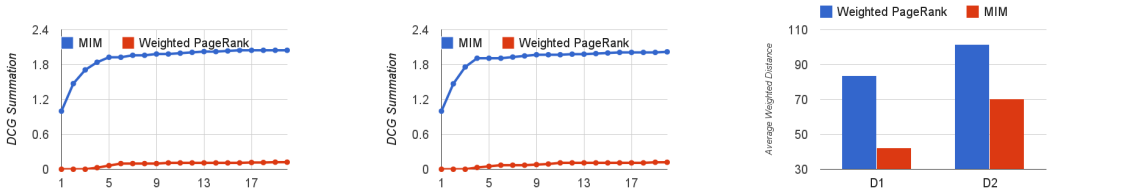
TABLE XVI. Top 10 Finance Companies

Yahoo! is contributing its talents to all others, and Facebook and LinkedIn are drawing talents from the rest of the companies. That phenomenon has been captured in our list. First, according to our result, Yahoo! is less magnetic than all the companies in the figure (except LinkedIn). Second, Facebook and LinkedIn already make themselves in the top 10 list, despite the fact that they are much younger and smaller than others. They have not outranked others due to other factors, since we are not only using talent traffic as it does in Figure 16.

Financial industry general trend. Table XVI lists the top 10 finance companies identified by PageRank and MIM results. J.P. Morgan Chase and Goldman Sachs are relatively unscathed by the recent financial crisis. That means the two companies stay attractive to talents because of their stabilities and capital power. Our model correctly identifies them as the top two. Goldman Sachs is widely recog-

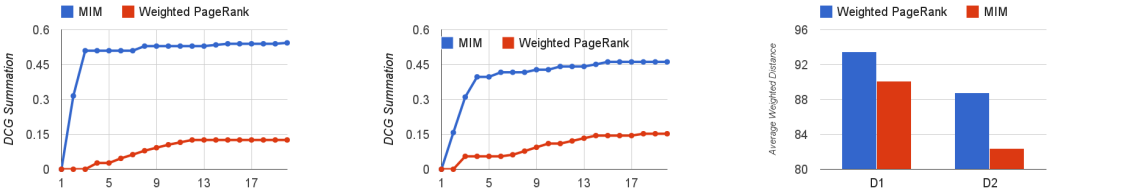
nized as one of the best financial companies, but PageRank fails to give it a proper rank. That's a clear advantage of our method in terms of magnet community identification.

We also have good diversity in the list in terms of segments and sizes. For example, we have leading companies of banks, real estates, financial services, and accounting. CB Richard Ellis and Jones Lang LaSalle are the well-known top 2 leaders in real estate segment. Blackrock is the leader in assess management segment. PWC (PriceWaterhouseCooper) is known as one of the "big four" accounting firms. Unlike IT industry where small start-ups may attract talents by their innovations, financial companies may be attractive due to their capitalization and customer relations, which makes smaller firms harder to stand out as magnet ones. Despite that fact, we also discover some smaller boutique ones in addition to big firms, such as Blackrock. Although it is a much younger and smaller company than others in the list, it has grown to be the world's largest asset manager and one of the leading investment management companies. PageRank's performance is similar to that of IT industry. It heavily prefers big cooperation, e.g., large banks, while ignoring the top companies in segments other than banking. Since big firms tend to be magnet ones in financial industry, PageRank's performance becomes better in financial industry compared to IT industry. We observe that in financial industry, the most admired companies and ideal employers survey results are not as consistent as that of IT industry. The reason for this phenomenon may be that, despite they have some basic functionalities in common, financial companies are more specialized than IT companies, which makes the attraction criteria vary more than that of IT.



(a) DCG on IT Ideal Employers (b) DCG on IT Admired Companies (c) Average Weighted Distance on IT

Figure 20. Performance on IT Industry



(a) DCG on Finance Ideal Employers (b) DCG on Finance Admired Corp. (c) Avg Weighted Dist. on Finance

Figure 21. Performance on Finance Industry

4.3.2.3 Overall Correctness measures

We use discounted cumulative gain (DCG (38)) to evaluate the ranking quality. It emphasizes the correctness of high ranked entities, which is the major goal of many information retrieval tasks. Figure 20(a) and (b) compare the marginal gain on DCG values at different cut-off positions on both ideal employer and admired company data in *IT industry*. As it shows, MIM outperforms PageRank significantly, because it identifies much more top-ranked companies correctly in the beginning. In the rank list generated by PageRank, the top-ranked companies are not the top in both datasets.

On aggregated level, we define a *weighted pairwise distance* measure of retrieved list L_1 and relevant list L_2, which emphasizes more on ranking order and correctness. $wDist(L_1, L_2) = \sum_{a_i \in L_2} (|L_2| - i + 1)(|i - \Lambda($ where $\Lambda(a_i)$ is the rank of a_i in L_1. Based on this definition, if an algorithm misplaces top-ranked entities, it will receive more penalties than another algorithm which misplaces low-ranked entities. Hence the smaller the value, the better the performance, Figure 20(c) shows MIM result has smaller weighted distance than PageRank result.

Similar to IT industry, we also examine the performance of MIM on financial industry. Figure 21(a) and (b) show its marginal DCG gain results comparing with PageRank. The performance is similar to IT industry, where MIM dominates PageRank performance in high-ranked company set.

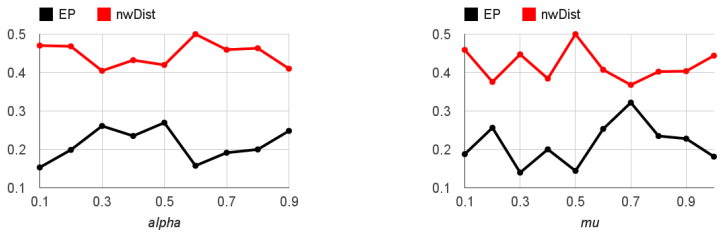
Figure 21(c) shows average weighted distance results. We notice that MIM does not outperforms as much as it does in IT industry data. We suspect two reasons for this phenomenon. First, the two benchmark data, i.e., survey results, are based on ad hoc criteria and not as diverse as MIM result. They always biased towards large corporations in banking and accounting. Moreover, some of their results are not consistent with people's usual understanding of the industry, e.g., US Bank ranks higher than

any other prestigious financial firms. If the benchmark needs calibration, the comparison results may be affected. Second, in financial industry, well-established ones tend to be magnetic automatically. In that sense, PageRank also does well because it identifies large companies. Even so, MIM still outperforms PageRank by a wide margin in terms of average weighted distance. This comparison between IT industry and financial industry clarifies that, when large communities are not necessarily the magnet targets, MIM is the ideal tool to discover magnet communities.

4.3.2.4 Parameter sensitivity

According to Eq. 7 and Eq. 8, the MIM algorithm has two parameters α and μ. Now we study how sensitive our model is to the two parameters. Due to space limitation, we only show the results on IT industry data and using admired company list as comparison. (Financial industry data give similar results.) Since we care more on high ranked entities, in addition to weighted distance, we also measure the model's performance on average precision $EP = \sum_{k=1}^{n} P(k)\Delta R(k)$, where $P(k)$ is the precision at cut-off k and $\Delta R(k)$ is the change of recall from position $k-1$ to k. We further normalize $wDist$ using $nwDist = \frac{1}{Z}wDist$, where Z is a normalization factor to make it align in the same scale as EP. In Figure 22 we can see that our model performs consistently on different α values. The fluctuations are in a small range. We also observe that the best performances are achieved at $\alpha = 0.6$.

Figure 22 also shows that μ has similar effects as α. Though the performance varies on different μ values, they are bounded by a small range. The highest average precision and smallest distance are achieved simultaneously at $\mu = 0.5$. Therefore, we assign 0.6 and 0.5 to α and μ to generate results of IT industry, and 0.2 and 0.7 to them for the same reason for financial industry in case studies.



(a) EP and $nwDist$ on α (b) EP and $nwDist$ on μ

Figure 22. Sensitivity Evaluation

4.4 Related Work

Network community analysis has been an active research field for a long time. However, previous works mainly focus on static or dynamic community detection, or most recently community evolution. To our best knowledge, there is no previous work directly related to *magnet community* identification.

Initially, people paid great attention on community detection via the structural properties of communities within a network, e.g., connection densities, etc (49). While majority of community detection has been done on static networks, new topics and methods addressing dynamic networks are proposed. Rosvall, et.al (73) used random walks to identify the structure of communities. Their work could also rank communities by their structural relations. However, their method also prefers large communities. Aggarwal, et.al. (5) captured the changing phenomena of groups of nodes in a network. They intent to track the gradual changes of one given community. Sun et.al. (76) identified communities that are locally static. The work in (81) identified communities with social interaction and their dynamics based an optimization model that minimizes social cost within communities. The work in (54) also tried to discover dynamic communities, such as growing or birthing communities from hidden member rela-

tions. Other studies focused on the discovery of inter-community structure using learning algorithms based on Markov process, which maximize the likelihood of interaction data (57)(91).

The goals of static and dynamic community detections are similar, which is to identify communities. They do not go beyond that to further analyze properties of identified communities, e.g., how magnet community is forming. Therefore, we are addressing a totally different problem where identifying community is only the starting point of our framework.

Another somewhat related direction is to discover the trend or properties related to community members. Falkowski et. al, tried to answer the question of how inactively participating a community would represent a user's interests (22). It focused on the fluctuation of members on a single community and the insights of the member-community interplay. Although our work also reflects members' interests about communities, it is about their selective attention over multiple communities. Furthermore, we are not emphasizing on the growth or demise of one single community, but the magnet interactions among multiple communities. Another work stood on a single node's (or a small subset of nodes') point of view to analyze how critical events affect *one* dynamic network (7). In our model, we take account of majority members' attentions rather than a small set of them to help determine the attractiveness levels of *many* communities. The authors in (43) studied the bursts of a short lived "community", or transient crowd, when a hot news emerges on a social network. Their work is mainly about detecting a special type of community of highly dynamic nature. In our case, the communities are not transformed that rapidly. Also, they are not considering anything related to *magnet communities*.

Regarding to our solution technique, constraint optimization formulation in link based graph data ranking has been proposed before with different intentions and design emphasis. The work in (14) con-

strains HITS algorithm (47) on user opinions about authorities. The work in (82) constrains PageRank algorithm on teleportation parameters to reflect personalization. The work in (3) learns parameters on transition matrix among nodes. The work in (4) tries to rank the whole set of nodes based on the known rankings of a small subset of them. The work in (27) proposes a more complete framework on utilizing meta data such as node and edge features and user preferences, which has never been completely considered in one work before. Our solution differs from the above on several points. First, our work is unsupervised. We don't have labels that indicate which communities are *magnetic*, or which one should absolutely has more attractiveness value than the other. Second, our constraint is derived on meta data, in addition to pairwise preferences. On the comparison with unsupervised approaches, e.g., (67)(42), our method considers more meta data such as node and edge features and magnet properties.

CHAPTER 5

MINING HIERARCHICAL RELATIONS IN PROFESSIONAL SOCIAL NETWORKS

5.1 Introduction

Powered by big data infrastructure, social network platforms are gathering data that reflect many aspects of our daily lives. One important aspect of such is people's relationships. For example, in physical world, there are varieties of relationships such as friends, colleagues, acquaintances, and they are also online in Facebook, LinkedIn, or Google+. Inferring different kinds of relationships can be critically beneficial to the social platform providers, for better understanding the users is the key for better services.

Responded by this trend, social relationship mining is becoming a promising research area. Although previous works have focused on different types of social relations mining, they mainly deal with *static* relations. Despite the fact that their data could be dynamic over time, the relation itself is static. Once it occurs, it would not change. For example, advisor-advisee relation (83) is static. Although a PhD student might change her advisor, the former advisory relation still remains. Family tree (84) is another static relation example. One's relation with his family members cannot change. In addition to static relations, there are also *dynamic* relations in social networks. For example, working relationship is dynamic. The manager-employee relation changes over time. Companies re-organize their organization chart *orgchart* constantly. The manager may leave, or the employee may switch manager, or their

relation may even reverse. The dynamic nature makes such relations much harder to infer than static ones. Figure 23 illustrates how the properties of dynamic and static relations are different.

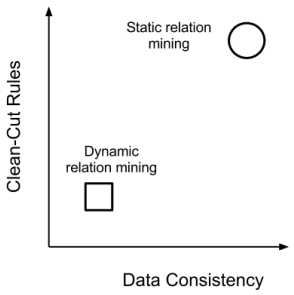


Figure 23. Property Differences of Dynamic and Static Relations

In addition to the dynamic nature, dynamic relations differ from static ones in two major ways. First of all, there lacks clean-cut rules for the relation inference. For example, unlike the publication records for advisor-advisee relation discovery, there is no strong signals for manager-employee relation identification. Table XVII lists the pairwise comparison of rules for an example of dynamic and static relation mining. In an academic network such as DBLP, there are certain rules that guide us design mining algorithms for the static relation. However, none of these rules work as they are in a professional network, such as Linkedin.

Consistency of the data is another new challenge for dynamic relation mining. If we only use the most recent data, there may not be enough information. However, if we use historical data, there will

	Advisor-advisee (static)	Manager-employee (dynamic)
connection	academic collaboration	voluntarily and adhoc
time	advisor has longer publication history than advisee	time to join a company does not imply job seniority
title	Professor/Student clear indicator of seniority	"code monkey" or "data guru" adhoc

TABLE XVII

STATIC AND DYNAMIC RELATION RULE COMPARISON

be inconsistency information with the current situation, such as company re-organizes itself. Static relation mining does not suffer from this difficulty. For example, historical publication records are indeed favorable information to infer advisor-advisee relation.

The two major challenges make pairwise dynamic relation hard to infer. Construction of hierarchical structure for the dynamic relations, such as orgchart of a company, is a even more challenging problem. Despite the challenges, such relation discovery is of great interest for industries, since many kinds of dynamic relations are greatly valuable. For example, the orgchart mining on Linkedin empowers better targeting for business to business advertisement, close friend or lover relations mining on Facebook will make the platform optimize its service to the users.

We are motivated by these challenges to design a framework for generic dynamic hierarchy relation mining, as well as implementing a system for one specific type of such relation, i.e., manager-employee relation on LinkedIn social network. First we try to integrate weak signals, e.g., vague connection indications, obsolete profiles, adhoc job titles, from heterogeneous sources to exhibit useful features. We proposed the concept of *objective entanglement group* and *subjective coherence strength* to capture

the dynamic relationship from objective and subjective perspectives. Second, these features are used to compute local probability of a dynamic relation. We design a local probabilistic function based on the physics principle of gravity. Finally, global dynamic relation probability is computed through a variation of Markov Decision Process (35).

In summary, our contribution lies in three folds.

- We are the first to tackle the dynamic hierarchy relation discovery problem in social networks. The novelty of the problem lies in two folds. First, dynamic relations on social networks are valuable and neglected by previous research. Second, unsupervised approach for hierarchy construction has never been done in the setting of dynamic relation on social networks. Although a few studies exist (62)(93), they are not applicable to our input and output, as well as rich social network features.
- We discover the connection between dynamic hierarchical relation mining problem and physics principle of forces. We integrate weak features from both subjective user generated social network features and objective opinions to mine dynamic relations to generate an effective local (or pairwise) dynamic relation predictor. We further formulate the global hierarchy optimization as a reduced Markov Decision Process problem, where the transit between states in a random walk represents the establishment of dynamic relation candidates. We also adopt a customized shortest path algorithm to solve the rMDP problem.
- We do the experiments on real world data sets from LinkedIn social network. We build a system for the mining task from raw data extraction to final result visualization. According to the experi-

ment results on two companies, our framework can produce accurate dynamic relation prediction. Therefore it is promising for real world applications.

The rest of the chapter is organized as follows. In section 2, we explain the difference between the dynamic relation and the static relation in details. In section 3, we will introduce the framework and signal integration. In section 4, we formally present the model. Section 5 is the experiment on real world social network data. Section 6 is the conclusion.

5.2 Related Work

Relation mining in social network is an active research field. Generally speaking, the major task of relation mining is to determine whether two entities have a particular relation (64)(19), and maybe the strength of such relation (90)(79). They do not deal with relations that further form a structure, like the orgchart in our case. We study a more specific problem than the broad works. The relation we are mining is *dynamic*, has *hierarchy*, and the framework is *unsupervised*. First of all, it is a well-motivated problem. In many real world social network applications, the relation is dynamic, e.g., the orgchart problem. Second, the relation has a multiple layer hierarchy, rather than a generic boolean relation. Third, the framework is good to be unsupervised, as labeling data is unavailable in many circumstances due to the resource limitation. Therefore, the generic relation mining studies do not have these unique constraints.

A few recent studies also focused on hierarchical relations, which is more specific than broad relation mining and closer to our work. For example, the work in (83) studied how to identify advisor-advisee relation in an academic collaboration network, and the work in (84) studied mining structured hierarchical relation, as family tree, in a social network data. Despite that they are closer to our work than others,

we are focusing on significantly different problems. First of all, our major focus is dynamic relation. In their cases, the relation does not change over time, so it is not dynamic. As Figure 23 and Table XVII indicate, they do not have the unique challenges we are facing. The rules in publication history (83) or family name relations and web forum post-commenting relation (84) are clear and strong to predict targeted relations. In our case, identifying ways to combine weak features to approximate rules is the first challenge that they do not have. Second, the dynamic relation is hierarchical and the framework is unsupervised. In paper (83), the relation is not hierarchical even though advisor and advisee relation is partial ordered. In paper (84), the relation is hierarchical, e.g., a family tree has multiple layers for different generations, but the framework proposed is supervised. In many real world applications, labeled data is unavailable. For example, in the LinkedIn social network data, although manager-employee relations may exist in the connection data, it will cost great amount of human labor to label them. In such cases, unsupervised method is more scalable and favorable. In (64), the authors tried to discover the roles of people in Enron company. In (79), the work tried to infer social ties of people. Although some of the relations they studied, i.e., online trust relation, colleagues, could be treated as dynamic ones, their focus was not the dynamic nature but different properties of the relationships, such as social ties (79), or the roles of different people in a particular event (64).

In a broader scope, there are other lines of research on hierarchy relation mining. In (62), the authors proposed a maximum likelihood approach to predict hierarchy in a social network. Different from dynamic relation, they have a static definition of hierarchy. For example, to extract the hierarchy in Bush administration, they use Google to retrieval the text form “X told Y” where X and Y are names in Bush administration dataset, and use the frequencies of such forms as interaction weights to define

hierarchy. Neither this form of data nor the hierarchy stability is applicable in our case. There are other studies on mining hierarchy from variety of data, not necessarily related to social network. In (93), the authors studied how to extract hierarchy from query terms, and the relation is artificially defined. In (52), a "meme-tracking" system is proposed to extract terms' evolution in blogsphere. Again, there is no dynamic relation issue there.

5.3 Solution Roadmap and Feature Extraction

5.3.1 Dynamic Hierarchical Relation Mining And Its Challenges

Dynamic nature is an important aspect of many kinds of real world social relations. The inference for such relation, especially with hierarchical information, is under explored. Existing social relation mining mainly targets static relations, or treat dynamic relations as static ones, as we analyzed in related work section. Comparing with static relation mining, there are three unique challenges with dynamic cases. First of all, dynamic nature makes it lack clean-cut rules to follow. Although manager-employee relation mining seems to be a similar task to advisor-advisee mining, all the clear rules in the static case cannot apply to the dynamic case, as it indicates in Table 1. Secondly, the data used for the relation inference may be inconsistent to the current situation. In static case, there is no such a challenge. For example, the papers published by a student and the advisor in the past is a strong indicator of their relation. However, in dynamic case, historical data may mislead the algorithm, since a manager-employee relation in the past may not exist in the present. Thirdly, in a dynamic case, the cleanness of the data is another issue. Take Linkedin social network as an example. People may not update their profile, e.g., job title, time to join a company, in time to reflect the dynamic changes of their working relations. In the static case, such as family relation, there is no out-dated data issue.

OBSERVATION (Dynamic Relation) In social networks, a relation is dynamic *when there is no guarantee that it cannot be changed. There are usually less clear rules to follow than the static relation mining, and there exists inconsistency in historical data to the current snapshot of the dynamic relation.*

The dynamic relation may have hierarchical structure, e.g., manager-employee relation can form orgchart. Due to the challenges, pairwise dynamic relation inference alone is already non-trivial. The more valuable hierarchical structure of such relation is even harder to imply. An analogy of this problem is the Phylogenetic tree construction, where the goal is to construct the evolutionary tree for species using the local inferences of ancestors (23). It is proved to be a NP-hard problem and brute-force method is computational infeasible. In the following subsections, we will derive and integrate features that are suitable for dynamic relation inference and propose an efficient algorithm for the global hierarchy construction.

5.3.2 Roadmap

Since the global hierarchy is very challenging to get, we divide the task into two steps: pairwise relation prediction and global hierarchy construction. The goal of the first step is to predict the local hierarchy as accurate as possible. The goal of the second step is to use the global information beyond local structure to optimize the local result so that the final result can be more accurate than the local one.

Pairwise Relation Formulation by a Classical Mechanics View When capturing pairwise relation, we want the model be able to quantify the following features.

- Asymmetricity. The impact from a manager to an employee should be larger than the impact vise versa.
- Attraction. The relation strength depends on both nodes' features.

- Distance. The relation strength depends on the distance between two nodes.

The insights of problems in social network research sometimes lie in physics. For example, the well-established independent cascade model (44) originated from particle interactions (55). This type of connection also exists in our case. If we treat nodes (social network users) as planets, the social network is an analogy of the universe. The interactions between nodes are analogous to forces between planets. We found that the dynamic relation between two nodes share common properties with the escaping phenomenon between a planet and a sun. First of all, they are both asymmetric. The escape velocities are different on whether it is a planet escaping form a sun or a sun from a planet. The chances of user A being B's manager is also different from B being A's manager. Second, they both depend on nodes' properties, whether it is mass difference or seniority or capability difference. Third, the distances among nodes, whether in the universe or social network, are effected by relations/interactions among them. Therefore, we borrow the concept of *escape velocity* to model the dynamic relation. We will first specify the way we derive features from the heterogeneous data, and then we use escape velocity to integrate the features for local pairwise dynamic relation prediction.

Global Hierarchy Construction. Getting the pairwise relation probability estimation is the starting point rather than the final result for the global hierarchy prediction, since the pairwise local probability for dynamic relationship only considers a target node and all its direct surroundings which is only the 1-hop relation. First of all, pairwise dynamic relation estimation is not accurate enough. For example, in the manager-employee relation discovery problem, one person usually only has one direct-report manager. For a person u and the manager candidate set Φ_u, if we choose $v = argmax_v p(v \succ u)$ as the final prediction result of u's manager, we would have many mis-assignment at the pairwise level. Further-

more, from the inaccurate pairwise level, the construction of the global hierarchy, e.g. the organization chart in our application, will be even further from the true structure. The global construction process helps in correcting local mis-prediction by global information.

We apply a bottom-up approach to construct the global hierarchical structure. For every node, we use a simplified version of markov decision process (35) to explore its path to the root of the hierarchy, i.e., the CEO or chairman of an orgchart. Each path represents a sequence of manager-employee prediction, which is a complimentary of the pairwise prediction using global structures. After such paths of all nodes have been explored, we choose the most frequent paths as the final hierarchical structure. Details about the algorithm will be introduced in the Ssection 4.

5.3.3 Feature Extraction

5.3.3.1 Objective Entanglement Strength

On a social network, dynamic relation is something that exists among users yet is not voluntarily marked by users. For example, a person may have many connections on a social network as LinkedIn with some of her colleagues. By observing the connections, one may not be able to tell which of her connections are those colleagues. However, the "public wisdom" may release their relations. For example, if the general users often view the person A and her particular connection B together, it is an objective indication that A and B probably share something in common. If they work for the same company, they may have fairly strong working relation or job similarity. Therefore, A and B are entangled together by the objective opinion of general users. Figure 24 is such an example. People who viewed Obama's profile on LinkedIn also viewed Romney's and Hirrary's profiles with high frequencies. The later two are

entangled with Obama by public opinion. Therefore, the first feature we pick up for inferring dynamic relation is *objective entanglement strength*.

Figure 24. Entanglement Example

Definition 5.3.1 *The Objective Entanglement Strength (OES) between entity x and y is defined as $\vartheta(x, y)$. It quantifies the objective similarity of x and y.*

Objective entanglement is a phenomenon that a small group of people being recognized as closely related to each other by the general crowd. Within an organization, if two people are recognized together more often than others, they tend to have more coherent relation. There are customized ways to quantify the OES for user pairs for different scenario. We use item-based collaborative filtering (74)(25) to generate "objective entangled pairs", where each such pair of users are highly entangled from objective perspective. Figure 25 gives an example of such. User x and y could be entangled as user u viewed their profiles together.

Within each pair, there are two persons that are frequently correlated to each other according to crowd opinions, e.g., profiles coviewed. The algorithm is similar to what is used in (74), except that we

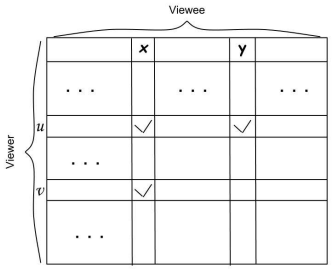


Figure 25. Entanglement Example

have a filtering function for the people with most frequent views. For example, in a social network as LinkedIn, Barack Obama or Bill Gates are among the top viewed individuals. Without filtering, this kind of people would appear in many people's objective entangled pairs, although they have little closeness. After filtering, we further restrict an entangled pair of people by the same community, e.g., a company in Linkedin, a group on Facebook, etc. Formally, the OES is defined as

$$\vartheta(x, y) = cos(\sigma(U_x(t)), \sigma(U_y(t)))$$

where $U_x(\tau) = \{c(i)e^{-(\tau-i)} | i \in (\tau - \omega, \tau)\}$ is the vector of counts of users $c(i)$ at each timestep i who have seen x's profile, and ω is the threshold controlling how long the historical data is. The formula of $U_y(\tau)$ aligns with that of $U_x(\tau)$, and σ is a filtering function [1]. There is a decay factor in the definition of $U_x(\tau)$ and $U_y(\tau)$ to weaken the information from a long time ago and strength the most

[1] $\sigma(v) = 0, if|v| > \theta$, where θ is a threshold on view frequency.

current information, which performs better in the dynamic relation discovery. The OES result could reflect relatively close working functions.

5.3.3.2 Subjective Coherence Strength

Besides objective clues, subjective ones are also valuable for dynamic hierarchical relation discovery. Subjective coherence of a group of people is the relation strength that is formed voluntary by themselves. When people connect to each other on a social network, they choose their connections carefully based on their subjective preference. Therefore, the connections reflect their real social relation to some extent. For example, on a professional networking scenario, people make connections based on their careers. If a group of people are well-connected, e.g., their connection forms a complete graph, they have higher chance to have similar job functions. If they are from the same company, they are probably coworkers. Figure 26 gives such an example. It is the connection graph of a company. Different colors mark different communities, which happen to have different job functions. Thus, people are "geologically" close to others who are their coworkers in this type of graph. Since the connection graph is subjectively formed, we use *subjective coherent strength* to measure how close people are.

Definition 5.3.2 *Subjective coherence strength (SCS) for a pair of node x and y is defined as $\varsigma(x, y)$. It quantifies subjective closeness between x and y.*

One way to capture the SCS between two nodes is to use the "distance" between them. Since each connection is subjectively formed, nodes distance is a natural measure of their closeness. However, we found that two most popular distance schemes, i.e., shortest path distance and commute distance (61) of random walk are not ideal for our case. First of all, in our problem settings, the edge is unweighted,

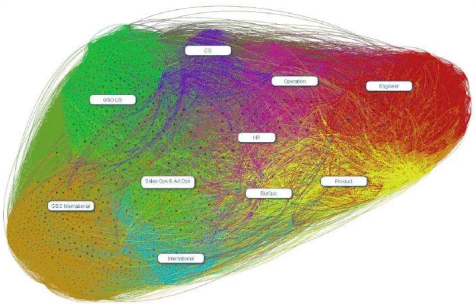
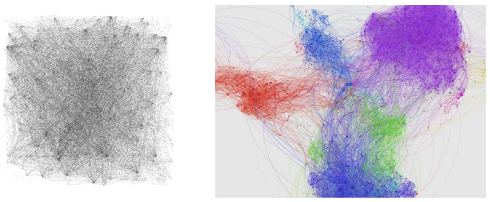


Figure 26. Relation Strength and Job Function

which makes a shortest path between two nodes just the count of hops between them. In a small-world graph, which is the phenomenon shared by majority of real world large scale graphs, even the diameter of the graph is usually a small number (88). Therefore, in our case, the shortest path is not a discriminative measure of the SCS. Second, the random walk commute time between to nodes in a *large* graph is proved to converge to a constant only related to the two nodes' degrees but not the global structure of the graph (61).

We use a force-directed algorithm which was firstly introduced by Yifan Hu (37) to compute the subjective coherence strength. The algorithm is initially proposed to visualize graphs and has been widely implemented in graph computation tools (63)(10). The objective of the algorithm is to come up with an arrangement of nodes and edges for a graph by modeling a physical system with forces acting between nodes, such that the whole system energy is minimized (see Figure 27 for a comparison with a random layout). The force-directed approach is a natural reflection of the underlining network structures as well as an efficient method that suits large graphs. We use the algorithm to produce pairwise node distances using euclidean scheme. In practice, such distance measure is a good indicator of the closeness

of working relation in our case, since the relative positions of nodes are optimized based on forces. It is finer-grained than shortest path distance, and reflecting more structural information than the random walk commute distance.



(a) random placement (b) force-directed placement

Figure 27. Layout Comparison

Another interesting fact to notice is that, if we throw in the whole company connection graph and get distance for everyone, it gives non-accurate results. In other words, people that are "far" from each other does not necessarily have different job functions. The reason for that is the whole graph distance calculation could dilute the force of surrounding nodes for every node. When considering the force among a node and its neighbors, many neighbor nodes are pulled away by their own neighbors/communities. To resolve this issue, we design a scheme to compute distances for nodes on their *ego network*.

Definition 5.3.3 *The ego network of node u is $\Omega_u = \{N_u, E_{N_u}\}$, where N_u is the set of adjacent nodes of u and $E_{N_u} = \{e_{ij} | i, j \in N_u\}$ is the edges among u's neighbors.*

For every node u in the social network, we first extract its ego network Ω_u. Secondly, we use Ω_u as input to the force based graph layout algorithm to get meaningful distance between u and its neighbors. Such distances are one type of representations of subjective coherence strength.

5.4 Model

5.4.1 Local Probability for Dynamic Relation Inference

Now we have extracted several features and discussed their pros and cons. Every feature that we have obtained can indicate the dynamic relation to some extend, yet not powerful enough to be any clear identifier. In this subsection, we try to integrate them in a more meaningful representation of the target dynamic relation.

5.4.1.1 Pairwise relation candidate selection

Even if connections have subjective coherence strength among each other, not every connection is a dynamic relation that we want to identify. Therefore, the first step is to refine the connections by selecting a candidate subset that really have great potential to have the dynamic relation we are looking for. Specified by our use case, we use user supplied job title similarity to do the candidate selection. Although, as we have discussed before, user supplied titles are noisy and inaccurate, there are some useful clues for us. For example, if one titles herself as “data scientist” and the other ”intern in data mining”, the two persons should have similar job functionalities, maybe even in the same team. Therefore, we also compute the pairwise job title similarity. We use the jaccard similarity measure for

the computation. Note that as the computation is one company at a time, instead of on the whole social network, it is efficient enough. The title similarity is defined as the following, the same as Jaccard index.

$$S(t_x, t_y) = \frac{|t_x \cap t_y|}{|t_x \cup t_y|}$$

where t_x and t_y are the sets of words in entity x's and y's titles. The dynamic relation candidate set Φ_u for node u is defined as follows.

$$\Phi_u = \{v | e_{uv} \in E, S(t_u, t_v) > \rho\}$$

where ρ is a threshold to control the title similarity to be higher than certain level.

5.4.1.2 Pairwise dynamic relation probability

Given u and $v \in \Phi_u$, we are ready to define pairwise dynamic relation probability $p(u \succ v)$. First of all, $p(u \succ v)$ is asymmetric. $p(u \succ v) \neq p(v \succ u)$. The reason is simple: the dynamic relation is hierarchical. For example, the probability that A is the manager of B is usually different from the probability that B is the employee of A, since the employee has only one manager and a manager has a group of employees. Note that the OES and SCS between u and v are also asymmetric. In other words, $\vartheta(u, v) \neq \vartheta(v, u)$ and $\varsigma(u, v) \neq \varsigma(v, u)$. The pairwise dynamic relation probability should be a meaningful integration of both subjective and objective features we have obtained. As mentioned in Section 3.2, we observe that the dynamic relation could be captured using physics principle about force. To draw an analogy, closer working relation, such as co-worker or manager-employee relation, can be interpreted as stronger force between two nodes. Escape velocity (11) is a suitable **representative** of

the force, as subjective coherence gives us relation "distance" and objective entanglement is an analogy for the gravity force attracting two nodes. For an object to "escape" another, the escape velocity is the minimum speed needed without further propulsion. If the velocity is large, it means the two objects are hard to separate, which is a good indicator for the close working relation. When we compute $p(u \succ v)$, i.e., the chance that u is v's manager, we could consider how much velocity v needs to leave u(see Figure 28). In other words, if v needs a large "velocity" to "escape" u, u has higher chance to be v's close co-worker or manager. The escape velocity also captures asymmetricity well, i.e., u's escape velocity from planet v is different from the vice versa.

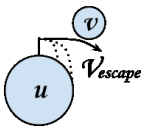


Figure 28. Pairwise dynamic relation

The classic form of escape velocity is

$$v_{escape} = \sqrt{\frac{2GM}{r}},$$

where G is the universal gravitational constant, M is the mass of the planet, and r is the distance from the center of gravity. In consistent with this form, we define the measure for the dynamic hierarchy relation as

$$V(x,y) = k\sqrt{\frac{\vartheta(x,y)}{\varsigma(x,y)}},$$

where k is a constant, $\vartheta(x,y)$ is like the relative mass of x to y, and $\varsigma(x,y)$ is the force-based distance we mentioned earlier.

Furthermore, we translate $V(x,y)$ into the range of $(0,1)$ to get the probability.

$$p(x \succ y) = \frac{2}{e^{-V(x,y)}+1} - 1$$

To address the issues, we design a global hierarchy construction algorithm. Our goal is two-folds. First, we want to correct the mis-arrangement of pairwise dynamic relation by utilizing global structure information. Second, in return of that, the global structure should have the benefit to be more accurate from the improved local prediction. Figure 29 demonstrates the reason that we could have the mutual benefits for local prediction and global construction. Imagine that the figure shows a part of an organization chart prediction in a company. The solid-line arrow means the manager prediction with highest pairwise probability, while the dot-line arrows are the prediction with other manager candidates. One thing to notice is that the solid-line arrow does not necessarily point to the real managers. Our goal is to correct the mis-predictions, e.g., the red arrow, using global structure which is not accessible when doing the local pairwise prediction.

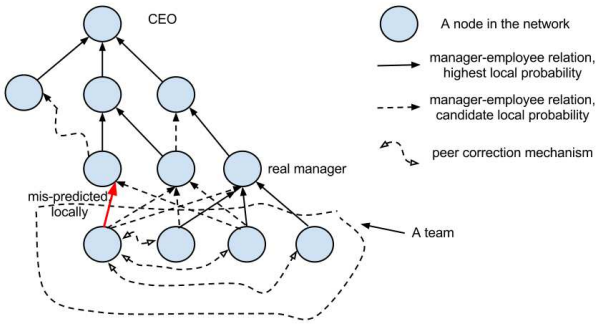


Figure 29. Global Optimization Illustration

5.4.2 Global Hierarchy Construction

As we mentioned previously, global hierarchy construction helps adjusting the whole hierarchy to correct local manager-employee assignment mistakes. There are two types of correction we formulated, with the help of additional information from the data. Type I correction is the *peer correction*. If a team is reasonably well organized in the raw data, it is very likely to be discovered by the local prediction scheme, as close co-workers tend to be close in the local results. If there is one mis-inference and majority other co-workers' manager are predicted right, we can correct the mis-inference. As it shows in Figure 29, the red arrow is likely to be reassigned given its co-workers' assignments. Type II correction uses more global structure information. We call it *efficiency correction*. The intuition is that in an efficient working environment, the paths from every employee to the CEO should be as short as possible. Therefore, we could measure the shortest path between a node and the CEO node. For example, locally, A may be closer to B than C. However, B is a lot further to the CEO than C. In a global

setting, A should choose C for a shorter path to the CEO node. The two types of corrections lead us to design the global structure construction algorithm.

When predicting hierarchy, title seniority is another important factor. For example, a person with a title "software engineer manager" is almost for sure more senior than a person as "software engineer intern". Therefore, after we identify close coworkers, such seniority is important to discover the hierarchy. However, it has its own drawbacks to prevent us from directly applying the seniority to predict the pairwise dynamic relation. The major problem is that such seniority is very noisy. For example, a person may have multiple jobs simultaneously. She could be a CEO for her own startup, while at the same time a manager in another big company she works for. Therefore, it is often inaccurate to directly infer the seniority from user supplied titles or positions, and it is also inaccurate to infer it from other source of information such as connections. Nevertheless, title seniority could still be an important feature. Like other noisy yet important features, we integrate title seniority with other information to make it a good component for hierarchy construction. Table XVIII lists the seniorities we organized.

5.4.2.1 Reduced Markov Decision Process rMDP Formulation

Intuition. From the local pairwise dynamic relation prediction result, we have a graph where each node represents a person and each edge connects a person and its manager candidate with a weight (prediction probability). Our mission is to derive a generated tree from the graph, which has the maximum likelihood to be the real hierarchy (in this case the orgchart with the CEO as the root of the tree). Given this setting, we can treat the hierarchy construction process as a random walk on the graph. Each *node (a person)* could be a *state* in the random process, and each *directed weighted edge* represents the *transit probability* between two states. Moving from one state to another means to establish

Seniority	Number
Owner or Founder	9
Executives	8
Vice President	7
Director	6
Manager	5
Associate	4
Entry Level	3
Intern	2
Unpaid	1

TABLE XVIII

SENIORITY LEVELS

a manager-employee relation with the transit probability. The *seniority differences* between two nodes can represent the *reward* for moving between the two "states". Therefore, the construction phase is converted to a random process with rewards. Markov Decision Process (35) (MDP) is a good model for capturing the problem. The general idea of MDP is a Markov chain with actions and rewards. It is a simulation of decision making process with random probabilities and the decision maker's actions. Customized in our case, the MDP is formulated as the follows.

Definition 5.4.1 *A Markov Decision Process w.r.t dynamic hierarchy relation inference is 4-tuple* (S, A, P, R)*, where*

- *S = N is the node set of the graph.*
- *A is a finite set of actions which we will discuss later.*

- $P = \{p(x \succ y) | x \in N, y \in \Psi_x\}$ *is the set of pairwise probabilities, which are also transitional probabilities between different states (nodes) in the MDP.*
- $R = \{r(x, y) | x \in N, y \in \Psi_x\}$ *is the set of reward functions. In our scenario, the reward of moving from state* x *to* y *is based on seniorities of* x *and* y*, which we will define later.*

In our case, the reward for assigning x's manager to be y is related to the difference of their seniorities. Ideally, one's manager's seniority is the smallest value that is larger than one's own seniority. However, in reality, the case is really different because of the data quality issue that we stated before. When one's seniority is mis-generated, there may be the case where a manager's seniority is the same, or even lower than the employee's seniority value. To tolerate such scenario and also capture the real value of seniority in our model, we define a reward function for assigning y to be x's manager as follows.

$$r(x, y) = e^{-\lambda((\omega_y - \omega_x) - 1)^2}$$

, where ω_x is the seniority value of x and λ is a constant factor. From the function we can see that $r(x, y)$ is maximized when y is 1-level senior than x. The reward value decreases sharply to other situations. Figure 30 illustrates the shape of the function.

The objective function of a standard MDP is the rewards maximize (35), given the transition probability P and action A. In the case of reduced MDP, we can formally write the objective function as

$$max \sum_{x \in N, y \in \Psi x} r_a(x, y)$$

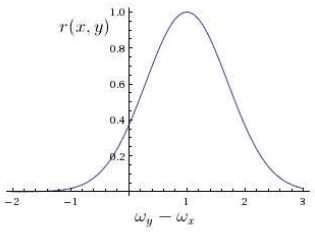


Figure 30. Reward Function

The *action* here has only one type. There is only one action through which a state transits to another, even though the rewards vary. In an ordinary MDP, there are usually more than one type of actions. The rewards of reaching the same state by different actions are different. However, in our case, there is no need for multiple type of actions. Therefore, we name our model as Reduced MDP, which is a simplified version with only one type of actions.

5.4.2.2 Shortest Path Solution

To solve a MDP, reinforce learning and dynamic programming are usually suitable tools (35). However, since we are facing a reduced MDP, we design a more efficient way to solve it. The algorithm involves two parts: merging the reward and transitional probability and computing single source shortest path.

Merge reward into transitional probability. As there is no explicit difference of actions, the chance of getting a reward $r(x, y)$ is to go along the path with probability $p(x \succ y)$. Therefore, the expected gain is simply $r(x, y)p(x \succ y)$. The real world meaning of this, in our application, is that by assigning y to be x's manager, the confidence is $p(x \succ y)$ and if the assignment is beneficial for the whole global

structure is $r(x, y)$. There is a trade-off w.r.t different combinations of rewards and probabilities. A high reward but low probability state should not be chosen if the expected value is low. Therefore, our objective is to generate a tree structure that maximizes the expected reward.

Single source shortest path Every time a pairwise dynamic relation is discovered, it contributes a little bit to the global hierarchy. In the organization chart mining problem, the final org-chart is built by all pairwise manager-employee relations. As we mentioned earlier, the structure is more efficient if it contains only the shortest paths from every employee to the CEO. Following this heuristic, combined with the merged reward and transitional probability approach, we could solve the reduced MDP problem using single source shortest path (SSSP) algorithm. In our case, the CEO node is the "source" in a company graph. With the expected gain as the weight of an edge, we can use SSSP to generate the shortest path tree which is the globally optimized hierarchy structure. The whole algorithm is illustrated in Algorithm 1.

5.5 evaluation

5.5.1 System Overview

Our framework involves several expensive computation tasks, such as large scale collaborative filtering and node distance calculation on large graph. The big data generated by real world social network application also make the computation more challenging. Therefore, we implemented the framework on the combination of MapReduce platform and traditional single machine paradigm. The whole system is illustrated in Figure 31. The expensive computations (modules in red border) are implemented on a Hadoop platform with 400 nodes. The rest modules are implemented in a single machine with 2.3GHz quad-core Intel Core i7 processors and 8GB RAM.

Algorithm 1 SSSP algorithm

Input: Target Graph $G(N, E)$; Pairwise dynamic relation probability P; Reward function R; Seniority Sn
Output: Global hierarchy $T \subset G$

// Initialization step
Compute the expected reward for every edge in G
// Start to compute SSSP from source node to all others
Set $i = 1$
while T is not converged **do**
  // Compute T_{i-1}
  Put the source in a queue Q and mark all other nodes to be unvisited
  while Q is not empty **do**
    Take out the head element h in Q and get all its neighbors in Q
    for v in h's neighbors **do**
      if reward(v,h)p($v \succ h$)¿Sn(u) + reward(u,h) p($u \succ h$) **then**
        update reward(v, h)
      end if
    end for
  end while
  //peer correction
  adjust u's assignment according to the assignments of Ψ_u
  i++
end while

5.5.2 Data Description

We use real world data from LinkedIn to test the performance of our framework. The dynamic relation here is hierarchical manager-employee relation. We use data from heterogeneous sources. The first type is the social network structure data. We extract intra-company social network, since it is only meaningful to infer working relations within the same company. Among all companies on LinkedIn.com, we choose two that we can get additional insights for the sake of evaluation, so that we could know the ground truth of the manager-employee relations. The basic statistics of our dataset is in Table XIX. The second type of data is behavioral data, more specifically, the profile view dataset. It contains user profile view events within half a year since the experiment date. All the data are from the heterogeneous

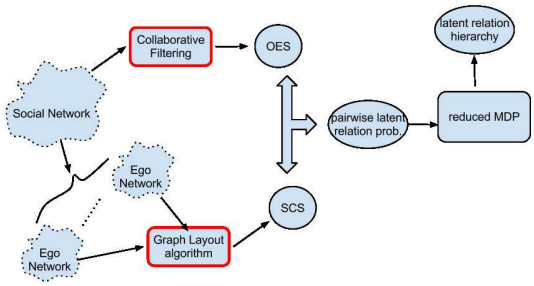


Figure 31. System Overview

information network. We will show the performance of our framework on the integration of such data and the prediction of the hierarchy dynamic relation.

	LinkedIn	EBay
#Nodes	3,518	26,116
#Edges	274,025	642,453

TABLE XIX

DATA DESCRIPTION

5.5.3 Experiment Result

5.5.3.1 Performance

Precision Measures. The precision measures on hierarchy structures are different from traditional metrics such as precision and recall. As it was demonstrated in (84), the predicted hierarchy structures can have various forms with huge quality differences but the same precision and recall. As few work has been done on evaluating such structural prediction qualities, the authors in (84) defined two new criteria. The first criterion is on ancestor sets comparison. In the predicted hierarchy, they find one ancestor set for every node in the hierarchy, and see if the set is a subset of the real ancestor set of the node. If the predicted ancestor set of one node is a subset of the real ancestor set, the overall performance increases by 1. The second criteria is on path accuracy. In the predicted hierarchy, they find the paths from every node to the root and compare them with the real path. Only when a predicted path matches the real path, it counts for one correct prediction.

In our opinion, the first criteria is too relax while the second one is too rigid. We propose a three-layer precision measure that is suitable to our application.

- P_P (*pairwise* accuracy). It is the same as the second criterion in (84). It is the most rigid one.
- P_C (*compatible* accuracy). If we did not assign a node to its real manager, but a close peer of the real manager, it still counts.
- P_L (*leapfrog* accuracy). If we did not assign a node to its real manager, but to the manager of the real manager, it still counts.

Candidate Set Coverage. The first thing we want to test is the size of manager candidate set size. After the pairwise dynamic relation prediction, we could have a set of manager candidates for every employee. The size of the set matters to the hierarchy construction phase. On the one hand, if the size is small, e.g., we only choose the candidate with highest pairwise probability, we may miss many real manager-employee dynamic relations. On the other hand, if the size is big, e.g., we include all connections with non-zero OES to be the candidates, we definitely waste much computation time. Figure 32 lists the relation between candidate set size and 3-layer accuracy coverage for C=3,5 and 7. Note that here we only consider pairwise local relation. P_P-coverage on $C = 3$ means the percentage of the real manager in the 3 top candidates with highest pairwise probability. Accordingly, P_C-coverage means the ratio of the cases when the real candidate or its coworker gets covered, and P_L-coverage means the real candidates or its real manager gets covered. From Figure 32, we can see that the gain of having more candidates is rather incremental than substantial. Besides, such coverage only considers whether the true candidate *exists* within the top-K rather than correctly making the assignment (which will be done in hierarchy construction phase). Therefore, we choose the candidate set size to be 3 in the rest of the experiments.

Hierarchy Structural Accuracy. Here we will test the effectiveness of our proposed method on the real datasets. Since there is no known applicable studies from previous work for the proposed dynamic hierarchy relation mining, we develop three simplified methods as baselines. As our approach integrates the features of objective and subjective for the local pairwise dynamic relation probability, we refer it as **SCE+OES**. The first baseline we compare against is only consider Objective Entanglement Strength (*OES*), namely the objective aggregated views from crowd users. It is a variation of the method in (74).

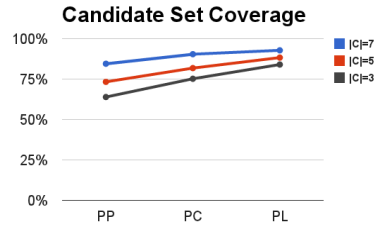


Figure 32. Candidate Set Coverage

The second baseline is to only use subjective coherence strength (*SCS*) to infer the dynamic relations. In other words, the dynamic hierarchy relations are constructed based on the nodes distances. Note that the both baselines also have the global hierarchy construction phase. It is that only the pairwise dynamic relation probabilities are different. Our third baseline is a straightforward one which we refer as naive baseline. It sorts candidates based on their seniority and SCS within the candidates cluster. The prediction made by the naive baseline is the one with 1-level higher seniority than the target node and strongest SCS.

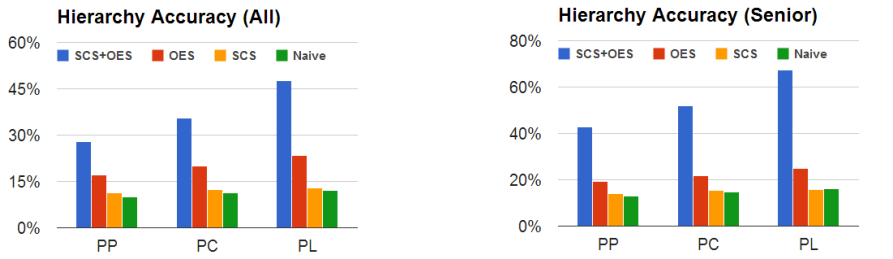
Figure 33 shows the hierarchy prediction accuracies on different scopes. We have it on overall accuracy in Figure 33(a) and accuracy focused on high-level positions with seniority higher than 6 (director and above) in Figure 33(b). Within each scope, the proposed method **SCE+OES** achieves the best performance compared with all the baselines. Specifically, it generally gives a $10-20\%$ improvement over the *OCS* method and a $10-30\%$ improvement over the *SCS* and naive baselines. This clearly demonstrates that neither standalone connection strength nor objective entanglement is good enough to infer hierarchy relations since the relations are dynamic in nature. Meanwhile, the

proposed framework can indeed combine a number of weak signals into meaningful features on building hierarchies and the proposed optimization method *RMDP* can accurately construct social relations on top of those features.

Another thing to notice is the different accuracies on different scopes of data. All kinds of precision measures are higher in Figure 33(b) than in 33(a). Although in real applications hierarchy accuracy is more important in higher-level prediction, we do not emphasize it in our algorithm. The reason for different accuracies lies in the data. From objective perspective, senior level people get more objective impressions than entry level people. For example, a company's CEO and CFO may be viewed a lot more than two interns in the same group. This phenomenon makes the OES more meaningful and accurate among high level managers, which may better lead our algorithm to discover their dynamic relations. From subjective perspective, senior level people have mature connections in terms of size, job functions, and quality. Entry level people tend to have less mature connections, majority of which are also entry level like themselves. This phenomenon also affects the SCS calculation. Last but not the least, usually a company's structure is more clear in higher level than entry level. Therefore, our algorithm has different performances given data with different qualities.

The accuracy is reasonable given the nature of our problem and the quality of data. We are the first to tackle dynamic relation mining in social networks and its hierarchy construction. The framework is designated to be unsupervised since labeled data for dynamic relation is often hard to get in reality. We notice that a closely related work (84), which focused on *supervised* learning of *static* relation, has precision on the same order of magnitude as ours. We did not implement their algorithm because it is not applicable (details in Related Work section), i.e., unsupervised v.s. supervised and dynamic v.s.

static, but at least we are confident with our performance. In real world applications, e.g., advertisement targeting or recruiting, one can always relax the precision requirement to P_C or P_L and still get good targeting performance.



(a) Overall Hierarchy Accuracy (b) Senior Level Hierarchy Accuracy

Figure 33. Hierarchy Structural Accuracy

Efficiency Results. As we demonstrated in Figure 31, some modules of the proposed framework are computationally expensive, such as collaborative filtering and graph layout generation. However, the heavy duty computations, processed offline by a Hadoop system, are only the pre-process and they are often the routine process of other real world application as well. For example, collaborative filtering process is a common task for other purpose on LinkedIn.com, and we only take its result piped into our system. For other modules in the framework, a single machine is capable to process in an online manner. The detailed running time results of various models are illustrated in Table XX.

	Time (sec)	
Module (offline)	LinkedIn	eBay
Collaborative filtering (result extraction)	238	156
Graph layout generation	3442	1924
Module (online)	LinkedIn	eBay
Pairwise latent relation probability	7	5
Global hierarchy construction	62	42

TABLE XX

RUNNING TIME BY MODULES

From the table, one can observe that the computation on graph layout generation takes most time. The generation is to compute subjective coherence strength every ego network. Although each ego network is not large, it is a sizable computation for a company with thousands of nodes. However, this process is just one routine for other site functions, so reduce the time cost of this part is out of scope of the current chapter, and it could be an interesting issue in the future work. We can further observe that the other components of key procedures run fast. Given the pipeline data from basic routine such as collaborative filtering and graph layout generation, the core process only takes around 1 minutes. Another insight is that the running time depends not only on the size of a social network, but also the density of its edges. LinkedIn has fewer nodes than eBay but more connection density. Therefore, the running time for LinkedIn company is longer than eBay.

5.5.3.2 Case Study

Before diving into aggregated level of performance analysis such as prediction precision, we first go over two case studies to illustrate the orgchart we discover and compare them with ground truth.

Linkedin Corp. - Dynamic Structure Discovery Figure 34 is a partial orgchart (executive level) of Linkedin Corp that is derived by our model. We use it to illustrate two advantages of our model, namely accuracy and dynamics. According to ground truth, we only have 1 mis-prediction (the red arrow), where we assign a director of web security team directly reporting to the CEO. The director is not truly in the executive team. However, this is indeed a showcase for **dynamic** hierarchy prediction. After we dig into the data, we found that the reason he shows up and gets directly assigned to the CEO is a security break on Linkedin happened in May, 2012 [1]. The director was leading a team resolving the issue and posting status update to the general public via Linkedin's official blog. This event increases the OES between the director and the CEO. It is the reason for this "mis-prediction". From another perspective, our method captures some dynamics in the orgchart according to the **dynamic nature** of our data and features.

eBay research lab. Figure 35 is a partial orgchart of eBay research lab [2] that is derived by our model. It is exactly the same with the real organization structure except that we have 1 person missing (the dotted node). The missing phenomenon is due to a constraint we made in the model. The way we quantify objective entanglement strength is collaborative filtering which essentially is based on profile

---

[1] http://blog.linkedin.com/2012/06/06/linkedin-member-passwords-compromised/

[2] http://labs.ebay.com

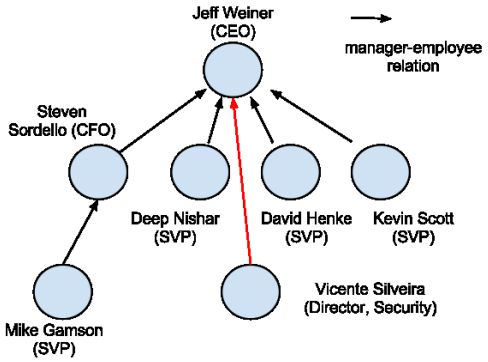


Figure 34. OrgChart Sample for Linkedin Executives

view data. If two people work together as a manager and an employee, but they never get coviewed by other people, there will be 0 OES between them. That means we would not consider the two for any candidate pair for the dynamic relation. This is the reason for having missing phenomenon as in Figure 35. The fundamental reason is the mismatch between real life relation and the data online. For example, two people can be very close friends, but they may not connect on Facebook. To handle the mismatch is another interesting research area, but it is out of the scope of the current chapter.

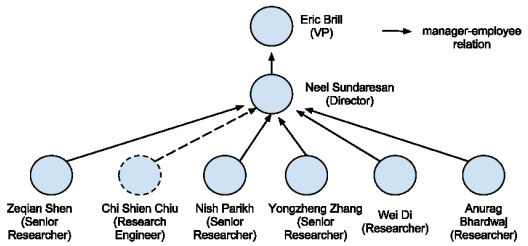


Figure 35. OrgChart Sample at eBay Research Lab

CHAPTER 6

CONCLUSIONS AND CONTRIBUTIONS

In this thesis, we have been exploring graph based approaches for social network mining. Towards this direction, we have studied several new research topics and proposed graph-based techniques for solving these new problems. We first invented the review graph as an instance of the graph based approach to tackle the online shopping review spam detection problem. By extending the graph based approach, we further explored the intertwining relation of influence and similarity of a heterogeneous network. Furthermore, we proposed magnet community mining project and identified communities of such property within a given network structure. We also present the organization structure mining application in the end. The contributions we made are summarized as follows:

- To fight spammers, we introduced a novel review graph model and an iterative reinforcement method that utilizes influences between reviewer, review, and store. Our work is the first to consider clues that are out of the box of single reviewer's behaviors. Our method demonstrated how the review graph information reflects causes of spamming and release important clues of different types of spammers. We proposed a natural way to compute *trustiness, honesty, and reliability* scores, and demonstrated their effectiveness of interpreting reviewer's veracity and store's quality. Comparing with existing work, our method identifies more subtle spamming activities with good precision and human evaluator consistency.

- We observe that often times there is a demand of reducing ubiquitous heterogeneous network to IR network. We also discover the benefit of analysis social influence and relevance for two types of network together. We design a method for IR network decoupling and a framework for influence expectation and relevance maximization to pursue such benefit. We demonstrated the real benefit we have by resolving social influence analysis and relevance analysis together from large scale real world data. We believe that analysis on IR network has promising future because social influence and relevance studies are two building blocks for many research interests, such as clustering, classification and recommendation.
- We introduced the magnet community identification problem. It is not only an important problem for social network research, but also a potential building-block for other analysis. We formulate the problem by graph ranking with heterogeneous information and constraints. Our model takes dependency and standalone features, as well as the magnet properties of communities to generate reasonable communities' attractiveness values. Tailored for this type of ranking, our model outperforms classical link based ranking algorithm, i.e., PageRank and its variation. This paper is only an initial step towards the new direction of research on social network communities. Possible future expansions include applying our framework to other heterogeneous network or analyzing business problems with company attractiveness as a starting point.
- We study an interesting problem of dynamic hierarchical relation mining on social networks. Unlike static relation mining, there are unique challenges in the dynamic case. Moreover, the hierarchy structure of the relation is also dynamic. As the first one to tackle this problem, we design a force-based model to integrate data from heterogeneous sources to compute pairwise

dynamic relation probability, and use rMDP to infer the hierarchy of objects. We evaluate the framework on the world's largest professional network with an application of orgchart mining. The experiment results demonstrated the effectiveness of the framework.

CITED LITERATURE

1. Fake reviews on the web...caveat emptor. money.cnn.com/2006/05/10/news/companies/bogus_reviews.

2. Resellerratings cracks down on thecellshop.net's review bribing. http://consumerist.com/2008/05/resellerratings-cracks-down-on-thecellshopnets-review-bribing.html.

3. A. Agarwal, S. Chakrabarti, and S. Agarwal. Learning to rank networked entites. In *KDD'06*.

4. S. Agarwal. Ranking on graph data. In *ICML'06*.

5. C. C. Aggarwal and P. S. Yu. Online analysis of community evolution in data streams. In *The 5th SIAM International Conference on Data Mining*, 2005.

6. Alex, Andrew, and Courtney. http://blog.topprospect.com/2011/06/the-biggest-talent-losers-and-winners.

7. S. Asur and S. Parthasarathy. A viewpoint-based approach for interaction graph analysis. In *KDD'09*.

8. A. Barabasi and R. Albert. Emergence of scaling in random networks. *Science*, October 1999.

9. A. Barua, P. Konana, A. B. Whinston, and F. Yin. Driving e-business excellence. *MIT Sloan Management Review*, 2001.

10. M. Bastian, S. Heymann, and M. Jacomy. Gephi : An open source software for exploring and manipulating networks. In *AAAI'09*.

11. R. R. Bate, D. D. Mueller, and J. E. White. Fundamentals of astrodynamics. 1971.

12. S. P. Boyd and L. Vandenberghe. Convex optimization. *Cambridge University Press*, 2004.

13. X. Carreras, L. S. Marquez, and J. G. Salgado. Boosting trees for anti-spam email filtering. In *In Proceedings of RANLP-01, 4th International Conference on Recent Advances in Natural Language Processing, Tzigov Chark, BG*, pages 58–64, 2001.

14. H. Chang, D. Cohn, and A. K. McCallum. Learning to create customized authority lists. In *ICML'00*.

15. W. Chen, C. Wang, and Y. Wang. Scalable influence maximization for prevalent viral marketing in large-scale social networks, 2010.

16. W. Chen, Y. Wang, and S. Yang. Efficient influence maximization in social networks. In *KDD*, 2009.

17. D. Crandall, J. Kleinberg, D. Cosley, S. Suri, and D. Huttenlocher. Feedback effects between similarity and social influence in online communities, 2008.

18. D. L. Davies and D. W. Bouldin. A cluster separation measure. 2, 1979.

19. C. P. Diehl, G. Namata, and L. Getoor. Relationship identification for social network discovery. In *AAAI'07*.

20. P. Domingbos and M. Richardson. Ming the network value of customers. In *KDD'03*.

21. L. Egghe. Theory and practise of the g-index. *Scientometrics*.

22. T. Falkowski and M. Spiliopoulou. Users in volatile communities: Studying active participation and community evolution. *Lecture Notes in Computer Science*, 4511.

23. J. Felsenstein. Inferring phylogenies. 2004.

24. J. Fleiss and J. Cohen. The equivalence of weighted kappa and the intraclass correlation coefficient as measures of reliability. *Educational and Psychological Measurement*, 1973.

25. B. S. G. Linden and J. York. Amazon.com recommendations: item-to-item collaborative filtering. 2003.

26. P. Ganesan, H. Garcia-Molina, and J. Widom. Exploiting hierarchical domain structure to compute similarity. 21, 2003.

27. B. Gao, T. Liu, W. Wei, T. Wang, and H. Li. Semi-supervised ranking on very large graph with rich metadata. In *KDD'11*.

28. L. Getoor and C. P. Diehl. Link mining: a survey. 7, 2005.

29. A. Goyal, F. Bonchi, and L. Lakshmanan. Learning influence probabilities in social networks. In *WSDM'09*.

30. Z. Gyngyi and H. Garcia-Molina. Web spam taxonomy, 2005.

31. L. Hamers, Y. Hemeryck, G. Herweyers, M. Janssen, H. Keters, R. Rousseau, and A. Vanhoutte. Similarity measures in scientometric research: The jaccard index versus salton's cosine formula. 25, 1989.

32. Q. He, B. Chen, J. Pei, B. Qiu, P. Mitra, and C. L. Giles. Detecting topic evolution in scientific literature: How can citations help? In *CIKM*, 2009.

33. S. Henry. Co-citation in the scientific literature: A new measure of the relationship between two documents. *Journal of the American Society for Information Science*.

34. J. E. Hirsch. An index to quantify an individual scientific research output. 102, 2005.

35. R. A. Howard. Dynamic programming and markov processes. 1960.

36. M. Hu and B. Liu. Mining and summarizing customer reviews. In *KDD*, 2004.

37. Y. Hu. Efficient, high-quality force-directed graph drawing. 10.1:37–71, 2005.

38. K. Jarvelin and J. Kekalainen. Cumulated gain-based evaluation of ir techniques. 2002.

39. G. Jeh and J. Widom. Simrank: A measure of structural-context similarity. In *KDD*, 2002.

40. N. Jindal and B. Liu. Opinion spam and analysis. In *Proceedings of the international conference on Web search and web data mining*, WSDM '08, pages 219–230, New York, NY, USA, 2008. ACM.

41. N. Jindal, B. Liu, and E. Lim. Finding unusual review patterns using unexpected rules. In *Proceedings of the 19th ACM international conference on Information and knowledge management*, CIKM '10, pages 1549–1552, New York, NY, USA, 2010. ACM.

42. T. Joachims. Optimizing search engines using clickthrough data. In *KDD'02*.

43. K. Y. Kamath and J. Caverlee. Transient crowd discovery on the real-time social web. In *WSDM'11*.

44. D. Kempe, J. Kleinberg, and E. Tardos. Maximizing the spread of influence through a social network. In *KDD*, pages 137–146. ACM Press, 2003.

45. M. Kessler. Bibliographic coupling between scientific papers. 14, 1963.

46. M. Kimura and K. Saito. Tractable models for information diffusion on social networks. In *PKDD*, pages 259–271. Springer, 2006.

47. J. Kleinberg. Authoritative sources in a hyperlinked environment. In *Journal of the ACM (JACM)*, 1999.

48. J. M. Kleinberg. Authoritative sources in a hyperlinked environment. In *SODA*, 1998.

49. A. Lancichinetti and S. Fortunato. Community detection algorithms: A comparative analysis. 2009.

50. J. Landis and G. Koch. The measurement of observer agreement for categorical data. *Biometrics*, 1977.

51. H. W. Lauw, E. Lim, and K. Wang. Bias and controversy in evaluation system. In *IEEE Transactions on Knowledge and Data Engineering (TKDE)*, 2008.

52. J. Leskovec, L. Backstrom, and J. Kleinberg. Meme-tracking and the dynamics of the news cycle. In *KDD'09*.

53. J. Leskovec, A. Krause, C. Faloutsos, C.Guestrin, J. VanBriessan, and N. Glance. Cost-effective outbreak detection in networks. In *KDD*, 2007.

54. J. Li, W. Cheung, J. Liu, and C. Li. On discovering community trends in social networks. In *WI-IAT'09*.

55. T. M. Liggett. Interacting particle systems. 1985.

56. E. Lim, V. Nguyen, N. Jindal, B. Liu, and H. Lauw. Detecting product review spammers using rating behaviors. In *Proceedings of the 19th ACM international conference on Information and knowledge management*, CIKM '10, pages 939–948, New York, NY, USA, 2010. ACM.

57. Y. Lin, Y. Chi, S. Zhu, H. Sundaram, and B. L. Tseng. Facetnet: a framework for analyzing communities and their evolution in dynamic networks. In *WWW'08*.

58. Z. Lin, I. King, and M. R. Lyu. Pagesim: A novel link-based similarity measure for the world wide web. In *Web Intelligence*, 2006.

59. L. Liu, J. Tang, J. Han, M. Jiang, and S. Yang. Mining topic-level influence in heterogeneous networks. In *CIKM'10*.

60. W. Liu, J. Janssen, E. Milios, N. Japkowicz, and Y. Zhang. Node similarity in the citation graph. 2006.

61. U. Luxburg, A. Radl, and M. Hein. Hitting and commute times in large graphs are often misleading. 2010.

62. A. Maiya and T. Y. Berger-Wolf. Inferring the maximum likelihood hierarchy in social networks. In *CSE'09*, 2009.

63. Mathematica. http://www.wolfram.com/mathematica/.

64. A. McCallum, X. Wang, and A. Corrada-Emmanuel. Topic and role discovery in social networks with experiments on enron and academic email. 2007.

65. M. McGlohon, S. Bay, M. Anderle, D. Steier, and C. Faloutsos. Snare: A link analytic system for graph labeling and risk detection, 2009.

66. A. Mukherjee, B. Liu, J. Wang, N. Glance, and N. Jindal. Detecting group review spam. In *WWW 2011 Poster*, 2011.

67. L. Page, S. Brin, R. Motwani, and T. Winograd. The pagerank citation ranking: Bringing order to the web. In *Technical Report, Stanford Digital Library Technologies Project*, 1998.

68. L. Page, S. Brin, R. Motwani, and T. Winograd. The pagerank citation ranking: Bringing order to the web, 1998.

69. S. Pandit, D. Chau, S. Wang, and C. Faloutsos. Netprobe: A fast and scalable system for fraud detection in online auction networks, 2007.

70. R. Pearl and L. Reed. On the rate of growth of the population of the united states since 1790 and its mathematical representation. *Proceedings of the National Academy of Sciences*, 1920.

71. A. Popescu and O. Etzioni. Extracting product features and opinions from reviews. In *EMNLP*, 2005.

72. M. Richardson and P. Domingos. Ming knowledge-sharing sites for viral marketing. In *KDD'02*.

73. M. Rosvall and C. T. Bergstrom. Maps of random walks on complex networks reveal community structure. *PNAS*, 2008.

74. B. Sarwar, G. Karypis, J. Konstan, and J. Riedl. Item-based collaborative filtering recommendation algorithms. In *WWW'01*, 2001.

75. C. Shi, X. Kong, P. Yu, S. Xie, and B. Wu. Relevance search in heterogeneous networks. In *EDBT'12*.

76. J. Sun, S. Papadimitriou, P. Yu, and C. Faloutsos. Graphscope: Parameter-free mining of large time-evolving graphs. In *KDD'07*, 2007.

77. J. Sun, H. Qu, D. Chakrabarti, and C. Faloutsos. Relevance search and anomaly detection in bipartite graphs. In *KDD Explorations 2005*.

78. J. Tang and J. Z. et. al. Arnetminer: Extraction and mining of academic social networks. In *KDD'08*.

79. J. Tang, T. Lou, and J. Kleinberg. Inferring social ties across heterogenous networks. In *WSDM'12*.

80. J. Tang, J. Sun, C. Wang, and Z. Yang. Social influence analysis in large-scale networks. In *KDD'09*.

81. C. Tantipathananandh, T. B. Wolf, and D. Kempe. A framework for community identification in dynamic social networks. In *KDD'07*.

82. A. C. Tsoi, G. Morini, F. Scarselli, M. Hagenbuchner, and M. Maggini. Adaptive ranking of web pages. In *WWW'03*.

83. C. Wang, J. Han, Y. Jia, J. Tang, D. Zhang, Y. Yu, and J. Guo. Mining advisor-advisee relationships from research publication networks. In *KDD'10*.

84. C. Wang, J. Han, W. P. Lin, and H. Ji. Learning hierarchical relationships among partially ordered objects with heterogeneous attributes and links. In *SDM'12*.

85. G. Wang, Q. Hu, and P. Yu. Influence and similarity on heterogeneous networks. In *Conference on Information and Knowledge Management*, CIKM '12, 2012.

86. G. Wang, S. Xie, B. Liu, and P. Yu. Review graph based online store spammer detection. In *Proceedings of the international conference on data mining*, ICDM '11, 2011.

87. G. Wang, Y. Zhao, X. Shi, and P. Yu. Magnet community identification on social networks. In *ACM SIGKDD*, KDD '12, 2012.

88. D. Watts and S. Strogaz. Collective dynamics of 'small-world' networks. 1998.

89. G. Wu, D. Greene, B. Smyth, and P. Cunningham. Distortion as a validation criterion in the identification of suspicious reviews. Technical report, 2010.

90. R. Xiang, J. Naville, and M. Rogati. Modeling relationship strength in online social networks. In *WWW'10*.

91. T. Yang, Y. Chi, S. Zhu, Y. Gong, and R. Jin. Detecting communities and their evolutions in dynamic social networks–a bayesian approach. *Mach Learn*.

92. X. Yin, J. Han, and P. Yu. Truth discovery with multiple conflicting information providers on the web. In *Knowledge and Data Engineering, IEEE Transactions on*, 2008.

93. X. Yin and S. Shah. Building taxonomy of web search intents for name entity queries. In *WWW'10*, 2010.

94. P. Zhao, J. Han, and Y. Sun. P-rank: a comprehensive structural similarity measure over information networks. In *CIKM*, 2009.

VITA

NAME: Guan Wang

EDUCATION:

B.S. in Information Security, University of Science and Technology of China, 2007.

PUBLICATIONS

- Guan Wang, Yuchen Zhao, Philip S. Yu, Shaobo Liu, Simon Zhang, "We know who you work for: Mining Hierarchical Relations in Professional Social Networks", *in submission*.
- Guan Wang, Yuchen Zhao, Philip S. Yu, "A Commodity Approach on New Label Classification, in submission
- Guan Wang, Qingbo Hu, Philip S. Yu, Mutual Relation of Trust and Influence in Heterogeneous Social Networks, in submission
- Guan Wang, Qingbo Hu, Phiilp S. Yu, Sharing Behaviors Analysis over Different Social Media, in submission
- Qingbo Hu, Guan Wang, Philip S. Yu, "Deriving Latent Social Impulses to Determine Longevous Videos", *in Proceedings of the 23rd International Conference on World Wide Web (WWW'14 poster)*.
- Yuchen Zhao, Guan Wang, Philip S. Yu, Shaobo Liu, Simon Zhang, "Inferring Social Roles and Statuses in Social Networks", *in Proceedings of the 19nd ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD'13)*.

- Qingbo Hu, Guan Wang, Shuyang Lin, Philip S. Yu, “Silence behavior mining on online social networks”, *Collaborative Computing’13*.
- Guan Wang, Qingbo Hu, Philip S. Yu, “Influence and similarity on heterogeneous networks”, *in Proceedings of the 21st Conference of Information and Knowledge Management (CIKM’12)*.
- Guan Wang, Yuchen Zhao, Xiaoxiao Shi, Philip S. Yu, “Magnet Community Identification on Social Networks”, *in Proceedings of the 18nd ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD ’12)*.
- Sihong Xie, Guan Wang, Shuyang Lin, Philip S. Yu, “Review spam detection via temporal pattern discovery”, *in Proceedings of the 18nd ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD ’12)*.
- Sihong Xie, Guan Wang, Shuyang Lin, Philip S. Yu, “Review spam detection via time series pattern discovery”, *in Proceedings of the 21st International Conference on World Wide Web (WWW ’12)*.
- Guan Wang, Sihong Xie, Bing Liu, Philip S. Yu, “Review Graph Based Online Store Review Spammer Detection”, *in Proceedings of the IEEE 11th International Conference on Data Mining (ICDM ’11)*.
- Guan Wang, Sihong Xie, Bing Liu, Philip S. Yu, “Identify Online Store Review Spammers via Social Review Graph”, *ACM Transactions on Intelligent Systems and Technology*.
- Fengjiao Wang, Guan Wang, Philip S. Yu, Why Checkins: Mining User Motivation on Location Based Social Networks, in submission

- Qingbo Hu, Guan Wang, Shuyang Lin, Philip S. Yu, Extracting Latent Social Impulse to Assess Online Videos' Longevity, in submission
- Shuyang Lin, Qingbo Hu, Guan Wang, Philip S. Yu, Understanding community effect on information diffusion, in submission